Principled Approach to Natural Language Generation

Vom Fachbereich Informatik
der Technischen Universität Darmstadt
genehmigte

Dissertation

zur Erlangung des akademischen Grades
Doctor rerum naturalium (Dr. rer. nat.)

vorgelegt von
Yevgeniy Puzikov, M.Sc.
geboren in Borisov, Belarus

Tag der Einreichung: 26. April 2021
Tag der Disputation: 22. Juni 2021

Referenten: Prof. Dr. Iryna Gurevych, Darmstadt, Germany
Prof. Ido Dagan, Ramat Gan, Israel
Prof. Claire Gardent, Vandoeuvre-lès-Nancy Cedex, France

Darmstadt 2021
D17

Puzikov, Yevgeniy: Principled Approach to Natural Language Generation
Darmstadt, Technische Universität Darmstadt
Year thesis published in TUprints: 2021
Date of the viva voce: 22.06.2021
URN: urn:nbn:de:tuda-tuprints-191154
URL: https://tuprints.ulb.tu-darmstadt.de/id/eprint/19115

This document is provided by TUprints,
E-Publishing-Service of the TU Darmstadt
http://tuprints.ulb.tu-darmstadt.de
tuprints@ulb.tu-darmstadt.de

Abstract

The research field of Natural Language Generation offers practitioners a wide range of techniques for producing texts from a variety of data types. These techniques find their way into various real-world applications and help many people to automate time-consuming tasks of text production in many areas.

At the moment, the design and evaluation of text generation approaches is largely empirical. Many systems are being developed to solve one particular task and work on a single data type, which makes it hard to compare the approach to any other technique and critically evaluate its performance. Some systems employ complex machine learning algorithms to learn rich data representations and perform joint modeling of the steps involved in the process of text generation. Such approaches offer an attractive trade-off between the development costs and output quality, but often lack transparency in terms of the reasoning about the behavior of the system. The number of the proposed approaches constantly grows, but the methodology lags behind and sometimes fails to solicit a better understanding of which approaches work, and the reasons for it.

In this thesis we present our view on the task of text production from a methodological point of view. We analyze the existent scientific literature, examine common text generation approaches and the established evaluation protocols. We further propose a principled view on the problem: we break it into components, examine their interaction and develop a set of recommendations which are envisioned to offer assistance during the design or analysis of a study. We further conduct a range of experiments to test this framework in several text generation tasks.

First, we show that task specification analysis sometimes allows one to solve the problem at hand with very simple techniques, without resorting to the complex machinery of advanced statistical learning methods.

We further demonstrate the potential of the developed framework to find discrepancies in the established evaluation protocols. We show that sometimes neither metric, nor conventional human evaluation is sufficient to draw conclusions about system performance. We demonstrate how a system can fit the data to achieve high automatic metric scores, while falling short in terms of actual output quality.

Finally, we use the framework to demonstrate how one can develop effective text generation systems without sacrificing the transparency of the inner working logic, making the developed systems both accurate and reliable.

Zusammenfassung

Das Forschungsgebiet der natürlichen Text-/Sprachgenerierung stellt Anwendern eine breite Palette von Techniken zur Generieren von Texten aus Vielheit von Datenarten bereit. Diese Techniken werden in konkreten Anwendungen benutzt und helfen vielen Menschen in vielen Bereichen bei der Automatisierung zeitaufwändiger Textproduktion.

Im Moment ist das Enwickeln und die Auswertung von Algorithmen zur Textegenerierung weitgehend empirisch. Viele Systeme werden entwickelt, um eine bestimmte Aufgabe zu lösen und mit einer einzigen Datenart zu arbeiten, was es schwierig macht, den Ansatz mit anderen Techniken zu vergleichen und kritisch seine Leistung zu bewerten. Einige Systeme verwenden komplexe Algorithmen des maschinellen Lernens, um umfangreiche Datendarstellungen zu lernen und gemeinsame Modellierung der Schritte, die am Prozess der Textgenerierung beteiligt sind, auszuführen . Solche Ansätze bieten einen attraktiven Kompromiss von Entwicklungskosten und Ausgabequalität, erlauben aber häufig nicht, das Verhalten des Systems zu verstehen und zu analysieren. Die Zahl der vorgeschlagenen Ansätze wächst ständig, aber die Methodik hinken hinterher und manchmal tragen diese nicht zu einem besseren Verständnis bei, welche Ansätze funktionieren und warum.

In dieser Arbeit präsentieren wir unseren Standpunkt zur Aufgabe der Textproduktion von einem methodischen Blickpunkt. Wir analysieren die existierende wissenschaftliche Literatur, untersuchen gemeinsame Ansätze zur Texterzeugung und etablierte Bewertungsprotokolle. Wir schlagen ferner eine grundlegende Sichtweise auf das Problem vor: Wir zerlegen es in Komponenten, untersuchen deren Wechselwirkung und entwickeln eine Reihe von Empfehlungen, die zur Unterstützung bei der Gestaltung oder Analyse einer Studie genutzt werden können. Wir führen eine Reihe von Experimenten durch, um dieses Framework in mehreren Textgenerierungsaufgaben zu testen.

Zunächst zeigen wir, dass die Aufgabenspezifikationen so ist, dass es manchmal das vorliegende Problem mit sehr einfachen Techniken zu lösen erlaubt, ohne Rückgriffe auf die komplexe Maschinerie des fortgeschrittenen statistischen Lernens zu benötigen.

Wir zeigen weiter das Potenzial des entwickelten Frameworks, um Abweichungen in den existierenden Bewertungsprotokollen zu finden. Wir zeigen, dass manchmal weder metrische, noch konventionelle menschliche Bewertung ausreichend ist, um Rückschlüsse auf die Systemleistung zu ziehen. Wir demonstrieren, wie ein System die Daten anpassen kann, um hohe automatische metrische Werte zu erreichen, während sie in Bezug auf die tatsächliche Ausgangsqualität stagnieren oder schlechter werden.

Schließlich verwenden wir das Framework, um zu demonstrieren, wie man effektive Texterzeugungssysteme ohne Einbußen bei der Erklärbarkeit der inneren Arbeitslogik entwickeln kann, wodurch die entwickelten Systeme genau und zuverlässig zugleich werden.

Acknowledgements

This dissertation would not exist without the support of many people. I would like to thank Prof. Dr. Iryna Gurevych for giving me the opportunity to conduct my research at UKP Lab, for her continuous support, and for her excellent feedback. I am grateful to Prof. Ido Dagan and Prof. Claire Gardent for offering valuable insights during my scientific journey, and for agreeing to be reviewers of my thesis.

My work was supported by several funding agencies: the German Research Foundation (DFG) under grant No. GU 798/17-1, the DFG-funded research training group "Adaptive Preparation of Information form Heterogeneous Sources" (AIPHES, GRK 1994/1), the FAZIT Foundation scholarship and the German Federal Ministry of Education and Research (BMBF) as part of the Software Campus program under the promotional reference 01IS17050. While writing the thesis, I was also supported by the STIBET Doctorate promotion funded by the German Academic Exchange Service (DAAD).

I would like to thank my dear colleagues Ilia, Jan-Christoph, Ji-Ung and Michael who shared the hardships of scholarly life with me. I deeply appreciate the help of Tristan Miller, Maxime Peyrard, Nils Reimers and Markus Zopf who provided insightful comments and suggestions that greatly assisted my research.

My gratitude also goes to Gabi and Judith, with whom I collaborated in the beginning of my research, leading to my very first publication.

I have greatly benefited from staying at Bloomberg L.P. in London for several months, and I want to thank Andy, Joshua, Minjie, Vittorio, and the rest of AI London Team for giving me this opportunity and an exciting time in London.

Finally, I would also like to express my gratitude to my friends and family who supported me at all times during my PhD.

Contents

CHAPTER 1

Introduction

One of the major traits of modern society is its reliance on information — people constantly produce and consume data. Instant messengers, automatic online translation between hundreds of language pairs, language learning applications, personalized voice assistants, digital marketing form an integral part of modern society. Information consumption grows every day, and this growth demands faster and more scalable methods of analyzing and generating new data.

Information exists in many forms, but the most widespread one is human language text. The research area that revolves around text production is called natural language generation (NLG). It is part of natural language processing (NLP), a multi-disciplinary research field that deals with processing human language data, and lies at the intersection of linguistics, computer science, and artificial intelligence (AI). NLG has been an active area of research for decades, but the recent advancements in statistical learning theory, pattern recognition and machine learning has made it possible for NLG to truly widen its scope and scale up to solving complex real-world problems, like machine reading comprehension (Cui et al., 2019; Zhu et al., 2019), open-domain question answering (Chen et al., 2017a; Liu et al., 2019b; Sun et al., 2018), machine translation (Clinchant et al., 2019; Chen et al., 2020a), and many others.

Recent surveys of the current state of NLG research show that the rapid growth of the practice-oriented side of the NLG community is outpacing the theoretical work necessary to analyze the available techniques and develop commonly accepted protocols (Van der Lee et al., 2019; Gatt and Krahmer, 2018; Amidei et al., 2018; Gkatzia and Mahamood, 2015). Our thesis is motivated by this research gap and is based on the three prominent research problems.

Task and data dependence of NLG systems Many of the existent NLG systems (especially rule-based ones) are task-specific by design (Reiter, 1994; Belz and Gatt, 2008). While they offer a unified view of the language generation process that is compatible with linguistically and psychologically motivated theories, they are not as efficient as data-driven methods. On the other hand, purely statistical systems are data-hungry: large amounts of high quality and diverse training data is a prerequisite for constructing and training robust statistical models (Koehn and Knowles, 2017; Hou et al., 2018; Simpson and Gurevych, 2019). Recent studies also suggest that data-driven

end-to-end systems fail to learn implicit relations between inputs, if they are not given a sufficient amount of training data (Li et al., 2016). Domain- and task-dependence in the case of statistical systems manifests itself as a problem of acquiring the right data. The inability of these systems to handle situations which deviate from those seen in the training data often leads to undesirable behaviour which is very difficult (if possible) to handle without making amendments to the corpora and retraining the models.

Black-box behavior of the popular approaches Current trend of using complex data-driven architectures to tackle the problem in an end-to-end fashion exemplifies the second problem: a difficulty of reasoning about the *decision-making process* of an NLG system. Even if the systems solve the task at hand, this is a big issue from both an engineering, and a scientific point of view, since a research objective is more about understanding how things work than about getting a high score in a competition. Moreover, unless one understands the technique, he or she cannot improve it further.

This issue raises the question of reliability of various NLG techniques and necessitates the development of more transparent approaches.

Uninformative evaluation Conventional evaluation methodology in the NLG community suffers from several drawbacks. Widely used automatic metrics lack strong correlation with human judgements (Scott and Moore, 2007; Reiter and Belz, 2009), but why this is the case is still an open research question. Furthermore, the reliability of the available automatic metrics becomes questionable as the recent studies find that very often they measure data- and system-specific properties of different approaches (Novikova et al., 2017b). Moreover, metric-based evaluation does not say anything about the real-world effectiveness of the system under evaluation. Rare exceptions (Reiter et al., 2003; Belz and Gatt, 2008) make it clear that a solid assessment of an NLG approach with respect to a certain task or downstream application is very important, but is rarely done.

Research questions These challenges give rise to the following research questions, which we approach in the current thesis:

(1) How to evaluate and compare NLG systems?
(2) How to develop reliable NLG systems and guarantee good average-case performance and at least bearable worst-case performance?

1.1 Contributions

In order to answer these research questions, we perform critical analysis of the existent NLG work, and propose a new conceptual framework to approach NLG tasks. We further use the framework in a series of experiments to prove its efficacy and develop new state-of-the-art computational methods of data-to-text and text-to-text NLG. The following lists provide an overview of these

contributions:

Contributions associated with RQ1:

- We developed a novel conceptual framework for the design and analysis of NLG studies.
- We demonstrated the efficacy of the framework in subsequent case studies.
- We showed how the framework can be applied to detect common problems in NLG experiment design.
- We compared architecturally different systems and showed that evaluation depends not only on the exact assessment methods, but also task specification aspects.

Contributions associated to RQ2:

- We showed how exploiting the patterns in the available data allows one to design more efficient data-driven systems.
- We developed and compared template-based approaches and neural architectures, and demonstrated that "reliable" does not necessarily mean "complex". We further connected this to the principles defined in the proposed framework.
- We designed and developed several NLG approaches with strong performance guarantees and transparent decision-making process.

1.2 Publication Record

Parts of this thesis have been published in international peer-reviewed conference proceedings from NLG-focused venues. We list these publications below and indicate the thesis sections which build upon them:

- **Puzikov, Yevgeniy**, and Gardent, Claire and Dagan, Ido and Gurevych, Iryna. (2019). Revisiting the Binary Linearization Technique for Surface Realization. In *Proceedings of the 12th International Conference on Natural Language Generation (INLG 2019)*. Tokyo, Japan. (Chapter 6)
- **Puzikov, Yevgeniy**, and Gurevych, Iryna. (2018a). BinLin: a Simple Method of Dependency Tree Linearization. In *Proceedings of the First Workshop on Multilingual Surface Realisation (MSRW 2018)*. Melbourne, Australia. (Chapter 6)
- **Puzikov, Yevgeniy**, and Gurevych, Iryna. (2018b). E2E NLG Challenge: Neural Models Vs. Templates. In *Proceedings of the 11th International Conference on Natural Language Generation (INLG 2018)*. Tilburg, the Netherlands. (Chapter 1, Chapter 4)

1.3 Thesis Outline

The remainder of this thesis is structured in seven chapters. Below we provide a brief overview of the structure of this document and the content of each chapter:

Chapter 2: "*Background*":
To provide more context and solicit a better assessment of our contributions, we critically analyze previous work in NLG, focusing on the task and data specifications, dominant approaches and established evaluation practices. We point out research gaps that lead us to the development of the proposed framework.

Chapter 3 "*Sanity Polygon Framework*":
In this chapter we describe the five main components of the NLG experiments: task, data, evaluation criteria, metrics and human evaluation experiment. We examine the relations between them and explain how these relations interact under common task requirements. We further motivate the need for a principled methodology which distills the available knowledge into a usable form. Finally, we introduce our attempt to solve this issue: a framework which contains principles and recommendations for guiding the design of NLG studies.

Chapter 4 "*Detecting Task Specification Issues*":
In this chapter we employ the proposed framework in a case study on automatic generation of restaurant descriptions. We describe the findings of our participation in the 2017 E2E NLG Challenge, where we use the results of data analysis to develop two conceptually different NLG approaches that achieve competitive performance in the task. We further show how the framework can be used to reveal flaws in the design of the task and evaluation setup.

Chapter 5 "*Detecting Evaluation Discrepancies*":
This chapter describes another case study on sentence compression, i.e. automatic generation of sentence summaries. We show how the framework can solicit a deeper analysis than the one done by conventional methods. We find that a higher-scoring system can produce worse-quality outputs, and trace the issue back to the relations which hold between the task components.

Chapter 6 "*Reliable NLG*":
In this chapter we use the proposed framework to develop NLG approaches that have strong performance guarantees and demonstrate a more reliable behaviour, compared to current state-of-the-art NLG methods. We describe our experiments in the surface realization task and prove the effectiveness of the framework with the results achieved by our approach.

Chapter 7 "*Conclusion*":
Finally, we summarize the main contributions of the thesis and outline potential future research directions, both from empirical and theoretical points of view.

CHAPTER 2

Background

This section provides a brief overview of the NLG task. We describe the common inputs NLG systems expect, the dominant NLG approaches, their advantages and limitations. We also examine how NLG systems are evaluated. As we familiarize the reader with the task, we also outline the research gap which we attempt to fill with this thesis.

2.1 Task Decomposition

Historically, NLG has been defined as the sub-field of NLP, which is concerned with the *construction of computer systems than can produce understandable texts in human languages from some underlying representation of information* (Reiter and Dale, 2000).

Language generation involves several distinct steps. In the NLG community the first decomposition of this process was proposed by Thompson (1977) who distinguished system components that are responsible for strategic decisions about *what to say* from those that are concerned with tactical decisions about *how to say it*. Consequently, NLG systems developed text planning and linguistic realization components (Dale and Mellish, 1998).

As language generation was gaining more attention in the research community, the decomposition was also getting more fine-grained. In the early 90s Reiter (1994) conducted a survey of the available NLG systems. He found that, despite the theoretical differences, the systems had a similar architecture in terms of the modules they divided the generation process into, the computations the modules performed, and the way the modules interacted with each other. The author called this the *consensus architecture*, and attempted to draw connections between the system design decisions and the psycholinguistic knowledge about how language is generated by humans. Subsequent work in the NLG research community followed up on that idea and added more distinct language generation steps (Reiter and Dale, 2000). The current version of this modular NLG paradigm includes the stages described below.

I. Content determination The purpose of this stage is to *extract relevant information from the input and produce a conceptual representation of the meaning of the output text*. In many situations the input information is not only sufficient to produce the target text, but also redundant:

- Generating a document summary entails dropping unimportant content and selecting only relevant information nuggets.
- Creating an image caption means attending to objects of some selected categories and ignoring the others.
- Sensor measurements might be coming from hundreds of devices, but only a few would be needed to generate a report for a human operator.

Naturally, the system needs to select important information and ignore the irrelevant parts. Once the important information is detected, it needs to be converted into some representation: logic forms, a set of attribute-value pairs, knowledge base triples, etc. The representations are usually very application-specific and depend on the communicative goal and target audience, but often take the form of a graph-like structure with concepts denoting the relevant entities (Reiter, 1994). An excellent disposition of a popular schema-based content-determination approach is given in (McKeown, 1985).

II. Text structuring The next step is *ordering the presentation of the selected content*. The design of this module depends heavily on the application specifics. Systems that generate short, one-sentence outputs do not require any complex text structuring. This is the case of image captioning or factoid question answering. However, in systems which output long texts, this module is responsible for the overall structure of the passage, including paragraph and sentence ordering. It also deals with discourse coherence and the right temporal order of the information presentation. For example, a system that assists in the decision-making process ideally would generate a sequence of suggestions going from the least to the most specific advice, similar to a decision tree (Portet et al., 2009).

To identify ordering of the output parts, early approaches used hand-crafted structuring rules encoding common-sense reasoning and expert knowledge (McKeown, 1985; Mann and Thompson, 1988; Hovy, 1993). Later work shifted to applying machine learning methods to rank possible orderings (Barzilay and Lee, 2004; Lapata, 2006). The benefit of the latter is that such methods can be used to determine the optimal ordering of any units, including messages, information nuggets, events, passages with varying difficulty (Barzilay et al., 2002).

III. Sentence planning Once it is clear what content is relevant and how to organize the output, the next step is to *build the linguistic structure of the target text*. Composing sentence-level information pieces from the information selected during the content planning stage is the task of the sentence planning module.

Many systems designed until a decade ago would map the semantic concepts and semantic relations extracted during the content determination step to sentences using what is called *deep syntactic forms*, an abstract linguistic representation that specifies content words and grammatical

relationships (Reiter, 1994). Early approaches relied on a dictionary of domain-specific entity ontology and incrementally replaced concepts with content words and phrases. Finally, the propositions would be grouped into clauses and sentences (McDonald, 1983; Meteer, 1991).

This module is rarely present in modern NLG systems, but the tasks it is responsible for are still completed in an indirect way, as part of the lexicalization process, whereby words and phrases are selected to communicate the information.

IV. Referring expression generation Because texts often describe entities and interactions between them, there is a need for a module which would *produce a description of an entity that enables the hearer to identify that entity in a given context*. This is the task of a referring expression generation (REG) component (Dale and Reiter, 1995; Reiter and Dale, 2000).

As an example, consider the visual scene describing a finite domain with three distinct objects d_1, d_2, d_3 (Figure 2.1). The objects are characterized by a set of attributes $A = \{type, clothing, position\}$, each attribute has an associated value. If d_1 is the target, then a REG system would select the relevant attributes and their values $\{type = man, clothing = suit\}$, and generate an entity description *the man wearing a suit*.

Figure 2.1: A visual scene describing a finite domain with three distinct objects. Given the target object (the man on the left) and a set of attributes, a REG component would select a relevant set of attributes and generate the object's description: *the man wearing a suit*. Example from (Krahmer and van Deemter, 2012).

Sometimes REG systems are provided with the lexicalized entity mention candidates; in this case lexical realization is not part of the problem specification (Belz et al., 2010).

REG is aimed to distinguish domain entities from each other. How the system further uses mentions of the entities depends on many factors: whether they have already been mentioned, if there are any entities with similar names, whether the entity is a living being or not, and so on (Belz and Kow, 2010). A common approach is to frame the problem as a classification task, and choose the smallest set of properties that will identify the target referent (Krahmer and van Deemter, 2012).

V. Surface realization (linguistic realization) The final step of a text generation pipeline is to *combine the selected words and phrases to generate a well-formed sentence*. This component's task is to order sentence constituents and generate the right morphological forms of the content words. Depending on the chosen grammatical constructions, the system might also need to add function words and punctuation marks.

Prior work explored a plethora of approaches to surface realization. When application domains are small and variation is expected to be minimal, realization can be done with template-filling methods (Reiter et al., 1995). The biggest advantage of such methods is strong guarantees of the quality of the output, since generation of ungrammatical structures can be easily avoided. When considerable linguistic variation is required, better results can be obtained by using a probabilistic grammar learned from large corpora (White and Rajkumar, 2009) or generating sentences from syntactic dependency structures (Filippova and Strube, 2009).

The NLG subtasks presented above constitute the classic stages of the NLG process. Prior work explored some variation to the task decomposition we described. For example, Gatt and Krahmer (2018) mentions the additional steps of *sentence aggregation* and *lexicalization*: the former merges several messages into one, the latter chooses words and phrases among alternative concept-to-word mappings. On the other hand, both can also be considered part of the sentence planning stage. When describing the consensus architecture, Reiter (1994) also mentioned *morphology* and *formatting* as distinct steps of the pipeline. Again, both can be viewed as steps of the surface realization stage. Ultimately, a crisp task decomposition might not be possible; the crucial point is to account for these steps when modeling the language generation process.

Having provided an overview of the NLG task, we move on to its input-output specification. While it is clear that the output of an NLG system should be some text, input types vary, depending on the application. The next section will briefly describe the variety of data that can be used as input for NLG systems.

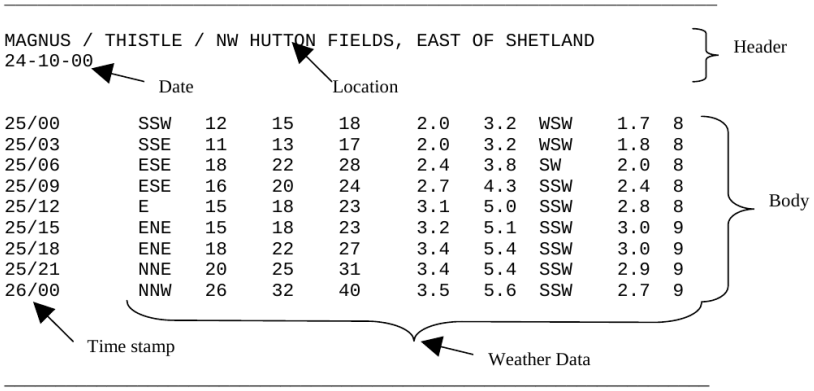
2.2 Input Data Types

Until recently, the most common input form was specially-formatted numerical data tables, but modern NLG research evolved beyond systems operating on non-linguistic data. Broadly speaking, all inputs fall into three main categories: non-textual (non-linguistic) data, syntactic or semantic text representations, and raw text.

2.2.1 Non-textual Data

Over the years, many NLG applications have been developed to automatically generate texts from numerical tables or knowledge bases. The main goal of such systems has been to *create a textual, human-readable description of the underlying datum which usually contains information not intended to be read by a human*.

For example, modern weather forecasting is largely based on computer simulations of numerical weather prediction approaches. These models predict time series with values of weather parameters (wind speed and direction, precipitation, etc.). Human forecasters use the time series data generated by the models as the major source of information when writing forecast texts. Sripada et al. (2004) proposed a data-driven approach (SUMTIME) to analyze the time series and automatically generate textual weather forecasts for oil company staff supporting offshore activities in the North Sea. The input to the system consists of several data files; for illustration purposes, we show one of such input files and an excerpt from the corresponding weather forecast in Figure 2.2.



(a) SUMTIME input

```
3.FORECAST 00-24 GMT, WEDNESDAY,        25-Oct   2000
  WIND(10M):   SSW 12-16 BACKING ESE 16-20 IN THE MORNING, BACKING
               NE EARLY AFTERNOON THEN NNW 24-28 LATE EVENING
      (50M):   SSW 15-20 BACKING ESE 20-25 IN THE MORNING, BACKING
               NE EARLY AFTERNOON THEN NNW 30-35 LATE EVENING
  SIG WAVE:    2.0-2.5 RISING 3.0-3.5 BY AFTERNOON
  MAX WAVE:    3.0-4.0 RISING 5.0-5.5 BY AFTERNOON
  WEATHER:     RAIN SOON, CLEARING TO SHOWERS IN THE EVENING
  VIS:         GOOD BECOMING MODERATE IN RAIN
```

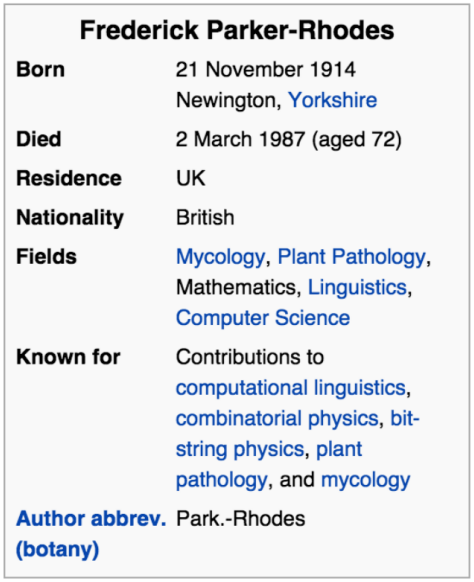
(b) SUMTIME output

Figure 2.2: An example of input and output of a data-to-text NLG system (SUMTIME). The input shows the predicted values for wind and wave related parameters at three hourly intervals. The bottom figure shows an excerpt from the weather forecast based on this data. Figures taken from (Sripada et al., 2003a).

The data used as input by NLG systems does not have to be purely numerical. Lebret et al. (2016) proposed a neural network-based approach to generate biographical sentences from fact tables extracted from the Wikipedia website.[1] The task requires the model to select among a large number of possible fields whose values vary from numbers to sequences of words (Figure 2.3).

In a similar vein, Wiseman et al. (2017) explored ways to create NBA basketball game summaries from the corresponding score tables. Similar to WikiBio, the ROTOWIRE and SBNation datasets used by the authors, contain a diverse mix of numerical and textual data (Figure 2.4).

[1] `https://en.wikipedia.org/`



Frederick Parker-Rhodes	
Born	21 November 1914 Newington, Yorkshire
Died	2 March 1987 (aged 72)
Residence	UK
Nationality	British
Fields	Mycology, Plant Pathology, Mathematics, Linguistics, Computer Science
Known for	Contributions to computational linguistics, combinatorial physics, bit-string physics, plant pathology, and mycology
Author abbrev. (botany)	Park.-Rhodes

Figure 2.3: Wikipedia infobox about Frederick Parker-Rhodes. The introduction of the Wikipedia article about him says: *Frederick Parker-Rhodes (21 March 1914 - 21 November 1987) was an English linguist, plant pathologist, computer scientist, mathematician, mystic, and mycologist.* Figure from Lebret et al. (2016).

Many applications of data-to-text generation techniques have been explored in the past. For example, Chen and Mooney (2008) developed a commentator system that was trained to follow a dynamic simulator state of a soccer game and generate commentaries. Iyer et al. (2016) showed how one can automatically generate high-level summaries of programming source code. Xu et al. (2015a) introduced an end-to-end approach to generating captions from images.

Usually numerical data assumes factual constraints on the content of the generated texts. However, depending on the properties of the training data, one might end up with a model exhibiting a large degree of output variation. This is dictated by the task specification: the generated text should match factual input data, but different words can be chosen to lexicalize the symbols.

2.2.2 Structured Representations of Text

Another large group of NLG approaches is concerned with generating texts from language meaning encoded in some structured text representations. The latter ones are obtained by processing raw text with semantic or syntactic parsers.

The first usages of semantic formalisms for statistical text generation go back to the days when researchers conceived the idea of developing language analysis systems using human brain as a model (Bateman, 1992). For example, given a sentence in *language A*, a human translator would read it, attempt to understand the meaning of it and then generate a sentence in *language B*. Similarly, grammar-based approaches to machine translation would parse an input first, thus

TEAM	WIN	LOSS	PTS	FG_PCT	RB	AS ...
Heat	11	12	103	49	47	27
Hawks	7	15	95	43	33	20

PLAYER	AS	RB	PT	FG	FGA	CITY ...
Tyler Johnson	5	2	27	8	16	Miami
Dwight Howard	4	17	23	9	11	Atlanta
Paul Millsap	2	9	21	8	12	Atlanta
Goran Dragic	4	2	21	8	17	Miami
Wayne Ellington	2	3	19	7	15	Miami
Dennis Schroder	7	4	17	8	15	Atlanta
Rodney McGruder	5	5	11	3	8	Miami
Thabo Sefolosha	5	5	10	5	11	Atlanta
Kyle Korver	5	3	9	3	9	Atlanta
...						

The Atlanta Hawks defeated the Miami Heat , 103 - 95 , at Philips Arena on Wednesday . Atlanta was in desperate need of a win and they were able to take care of a shorthanded Miami team here . Defense was key for the Hawks , as they held the Heat to 42 percent shooting and forced them to commit 16 turnovers . Atlanta also dominated in the paint , winning the rebounding battle , 47 - 34 , and outscoring them in the paint 58 - 26.The Hawks shot 49 percent from the field and assisted on 27 of their 43 made baskets . This was a near wire - to - wire win for the Hawks , as Miami held just one lead in the first five minutes . Miami (7 - 15) are as beat - up as anyone right now and it 's taking a toll on the heavily used starters . Hassan Whiteside really struggled in this game , as he amassed eight points , 12 rebounds and one blocks on 4 - of - 12 shooting ...

Figure 2.4: An example data-record and document pair from the ROTOWIRE dataset. Taken from Wiseman et al. (2017).

obtaining a semantic "footprint" of the original sentence. An output would then be generated, based on the rules of the grammar induced from training data, a semantic bank. An example of such a formalism is abstract meaning representation (AMR) (Knight and Luk, 1994; Banarescu et al., 2013) which treats sentences as atomic meaning-bearing units and uses graph structures to encode the meaning of each sentence (Figure 2.5).

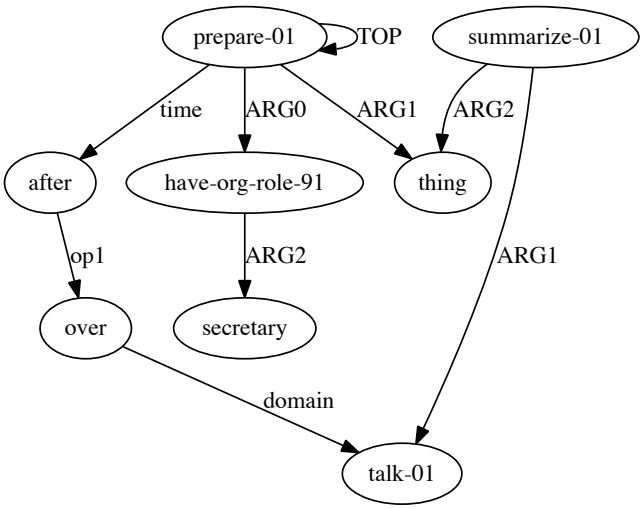


Figure 2.5: An AMR graph for the sentence: *After the talks were over, the secretary prepared the summary of the talks.*

AMR graph representations are powerful enough to capture complex semantic phenomena that are abundant in human language. They include entity identification and typing, semantic roles, entity grounding via wikification, as well as treatments of modality and negation. The potential

of AMR representations for generation has been explored in the SemEval-2017 Task 9 Generation Subtask (May and Priyadarshi, 2017).

AMR and similar high-level semantic formalisms encode meaning, and abstract away from the linguistic phenomena which do not carry much semantic information. For example, despite the differences in surface realization, the following sentences have the same AMR structure (Banarescu et al., 2013):

He described her as a genius.

His description of her: genius.

She was a genius, according to his description.

Unfortunately, semantic analyzers and/or training data for them are not available for many languages and language domains. In practice, the complexity of such formalisms also results in low annotator agreement scores, which ultimately inhibits the performance of semantic parsers trained on the annotated semantic banks.

As a result, some researchers proposed to rely on shallow linguistic knowledge (part-of-speech (POS) tags, syntactic trees, coreference chains) that can be automatically derived from a corpus, and bypass the step of semantic analysis (Mani et al., 1999; Barzilay and McKeown, 2005; Wities et al., 2017). The main motivation behind these approaches is that input sentences themselves contain enough information to re-generate text from a compressed structure that is composed of these lexicalized elements. Dependency grammar is, perhaps, the most wide-spread formalism that is used as a source of shallow linguistic knowledge (Figure 2.6).[2]

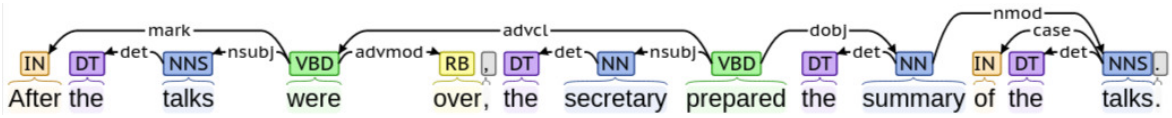


Figure 2.6: A dependency tree for the sentence: *After the talks were over, the secretary prepared the summary of the talks.* Compare with Figure 2.5.

Recently, surface realization, the task of generating texts from syntactic dependency tree representations, gained a lot of interest in the research community. Two shared tasks were organized to examine the potential and limitations of syntactic structures for generation (Belz et al., 2011; Mille et al., 2018). In order to better understand the importance of various components of a sentence structure, the organizers developed two different representations with a varying level of abstraction. We also participated in this task and will provide more details about our experiments in Chapter 6.

Following a similar line of work, some researchers attempted to develop other text-based representations; examples include Open Information Extraction (OpenIE) (Etzioni et al., 2008), Open

[2]Visualization done using the Stanford CoreNLP tool: `http://nlp.stanford.edu:8080/corenlp/process`

Knowledge Graph (Wities et al., 2017), and others. Such formalisms are suitable for extracting relevant information from documents without resorting to deep semantic analysis. The resulting representations are more scalable than dependency tree annotation and more consistent than semantic formalisms, since they do not require manual corpus annotation. More importantly, they carry a lot of surface linguistic information which helps to ensure grammaticality of the output texts. An example of a sentence with annotated dependency relations and extracted Open Information Extraction triples (Etzioni et al., 2008) is given in Figure 2.7.

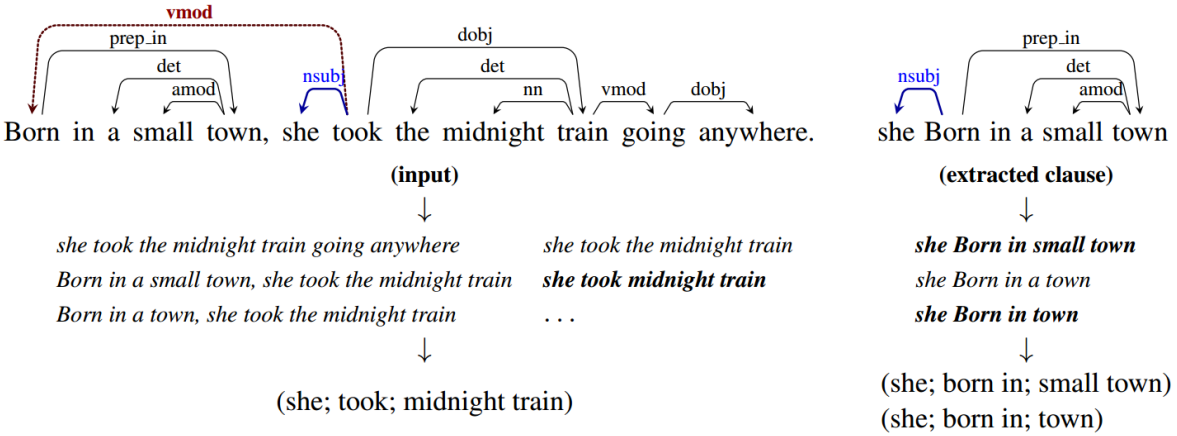


Figure 2.7: A sentence with annotated dependency relations (top left) and Open IE triples (bottom) extracted from it. The figure is taken from Angeli et al. (2015).

2.2.3 Raw Text

Generation from any structured representation has a weakness: one needs to make sure that the input *accurately and sufficiently represents the meaning that the generated text should convey*. In practice, this is very hard to achieve for several reasons. First, none of the available NLP tools achieves a 100 % accuracy in real-world applications. Consequently, any structured representation contains errors which propagate further down the NLP pipeline and find their way into the NLG system output. Second, even if the input representation would be perfect, in many tasks it is not even possible to generate the "right" output, since human languages are ambiguous by nature. The more abstract and underspecified the input representation is, the more freedom it permits for a generation system, and the harder it is to evaluate the generated text.

Text-to-text generation approaches are distinguished from those operating on structured textual representations, because they *take texts as input, and automatically produce new texts as output* (Gatt and Krahmer, 2018). Bypassing the need to create intermediate representations is very appealing, but until recently there have been no machine learning approaches that could process raw texts in a computationally efficient way. Researchers were focusing more on designing linguistic formalisms which would be complex enough to capture the variety of language phenomena, yet simple enough to allow consistent human annotations. An additional requirement for such

formalisms was the possibility of incremental processing of text segments. Otherwise, computationally it would be prohibitively expensive to use such a formalism in an application. This was the main reason why many NLG approaches of the past were application-specific: one often needed to re-engineer the system to make it work with inputs that were structurally different from the original ones.

The popularity of encoder-decoder neural approaches (Sutskever et al., 2014; Cho et al., 2014c) has caused an interesting paradigm shift: the community moved away from text-generation approaches that are designed to satisfy one particular communicative goal. Similar to the distinction between text-to-text and data-to-text systems, there used to be clear boundaries between NLG and, for example, document summarization, machine translation or question answering. However, modern encoder-decoder models can be applied to any textual data. This makes NLG methods and approaches generally applicable across various areas of NLP. Example applications that generate new texts from existing texts include machine translation (Sutskever et al., 2014; Bahdanau et al., 2015), sentence compression (Filippova et al., 2015; Zhao et al., 2018), document summarization (Zhou et al., 2018; Dhingra et al., 2017; Cui et al., 2017), text simplification (Vu et al., 2018; Wubben et al., 2012; Nisioi et al., 2017), question answering (Chen et al., 2017b; Tang et al., 2017), etc.

So far, we introduced common NLG subtasks, and described the most common ways to represent inputs to NLG systems. The next section will provide a brief overview of how NLG tasks are modeled by various approaches, and discuss some of the advantages and disadvantages of the common NLG techniques.

2.3 Dominant Approaches

2.3.1 Rule-based and Data-driven

There is no generally agreed topology of NLG systems. The first property that largely determines the type of a system is the *inference engine*: according to this criterion all systems can be divided into rule-based, statistical (or data-driven), and hybrid.

Rule-based Rule-based systems rely on a set of deterministic rules, either hand-written by experts, or extracted from large corpora (Dale et al., 2003; Reiter et al., 2005; Green, 2006; Galanis and Androutsopoulos, 2007). Usually, template-based systems are also categorized as rule-based, because templates are filled with content using rules.

Data-driven Statistical approaches are based on probabilistic models which are trained using large corpora. The latter are used as a source of valuable statistical patterns which data-driven models "learn" by undergoing some optimization procedure. These systems solve the same subtasks as rule-based approaches, but tackle them in an automatic way. Prior work explored the potential of statistical methods for modeling each individual component of the NLG pipeline in

isolation (Duboue and McKeown, 2002; Barzilay and Lapata, 2005, 2006), as well as simultaneously, by utilizing hierarchical generative models (Liang et al., 2009).

Hybrid Hybrid architectures combine rules and statistical learning. As mentioned in Section 2.1, NLG is a complex decision-making process, some parts of which could be effectively tackled with rules. A learning approach in some cases could be risky; constraining the possible "paths" along which the decision-making process should evolve can be safeguarding the system from getting into an unknown zone. Other decisions which permit a large space of possible candidates, on the other hand, is a perfect application for statistical approaches.

One of the earliest hybrid NLG models was the *overgenerate-and-select* system of Langkilde and Knight (1998): a rule-based symbolic generator produced a word lattice of possible renderings, and a statistical extractor chose one of them as the output string. A more recent example is the micro-planning system of Gardent and Perez-Beltrachini (2017) which jointly segments the text into sentences, aggregates them and performs surface realization. The overall architecture is based on a handwritten grammar which specifies grammatical structures, while subsequent steps are handled by data-driven components.

In practice, the choice between rule-based and data-driven techniques is determined by many factors. In restricted domains, where the set of rules is known in advance, it is generally safer to rely on hand-written templates, because the output quality has strong guarantees. An important advantage of templates is that they sidestep linguistic decision-making, and avoid the need for large complex knowledge resources and processing (Langkilde and Knight, 1998). However, many NLG systems operate under open-domain conditions; in such cases, it is practically impossible to codify all possible situations a system might encounter.

Statistical approaches have an advantage of being more scalable. Porting a rule-based system to another application requires significant engineering efforts, while data-driven systems can be re-trained on domain-specific training data. In cases where the in-domain data is very scarce, one has an option of using transfer learning techniques which leverage the data for a different task or domain in order to boost the system performance (Bozinovski and Fulgosi, 1976; Pratt, 1993; Pan and Yang, 2010).

However, data reliance is a double-edge sword: if training data is noisy, a data-driven algorithm is likely to fit to the noise as well, which will cause the system to generate erroneous outputs. Finding the source of errors is also not that easy: it could be the noisy data, deficiencies of the algorithm, or buggy system implementation. Some of the more advanced statistical approaches have very complex learning dynamics, which often leaves one no choice, other than to empirically validate the suitability of the method in a particular setting.

Hybrid approaches have the advantages of both rule-based and data-driven systems. They constrain the search space of possible candidates using rules, and allow for statistical learning with little training data. However, currently they do not seem to be very popular in the NLG community. One possible reason for this could be that hybrid systems require significant knowledge of both linguistic formalisms, and statistical learning theory, which is a rare mix of expertise to find in modern NLG community.

2.3.2 Modular and Joint

The second common classification criterion is the *subtask composition* which defines a system as modular or joint. This division is more interesting, because the decision of which stages to model explicitly has long-term implications not only for the performance of the system, but also its portability and extensibility.

Modular architectures Such approaches follow a linguistically motivated pipeline architecture with several steps and modules incorporating different subsets of the tasks described above.

Early approaches to the task of text generation adopted a unidirectional information flow: the system components were arranged in a linear order, with each component receiving information from its predecessor and sending information only to its successor (Reiter, 1994). Figure 2.8 shows an example of such a system — BT-45 which generates textual summaries from numerical and symbolic data describing the physiological condition of babies in a neonatal intensive care unit (Portet et al., 2009).

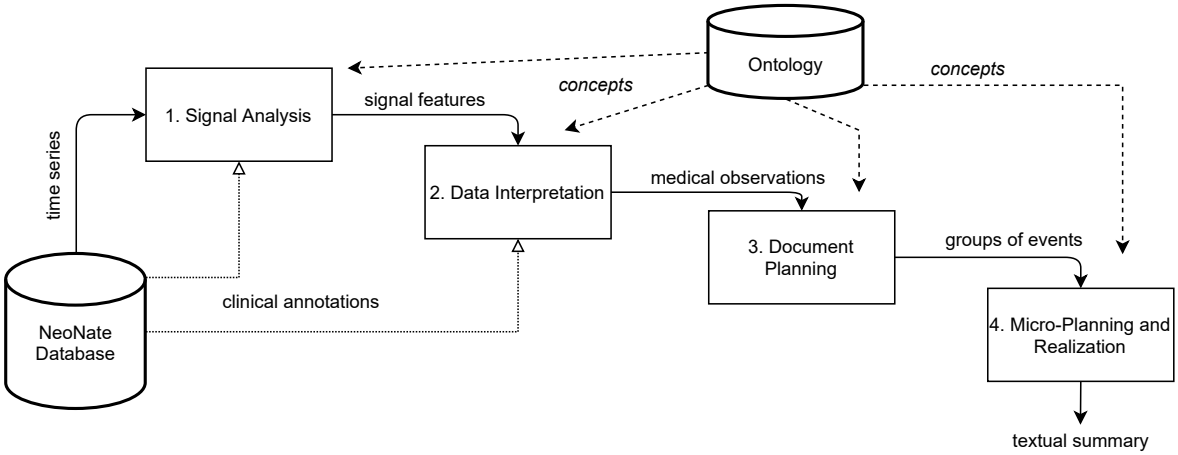


Figure 2.8: A schematic view of a modular NLG system: BT-45 (Portet et al., 2009), a system which generates textual summaries from physiological time series. Figure adapted from (Portet et al., 2009).

Some modular systems allow backward feedback loops or select pairs of components that are allowed to interact, but overall the systems remain pipelines. In most cases this is a deliberate design decision conceived to simplify the engineering effort of system development. NLP researchers generally agree that there are linguistic phenomena that can only be properly handled by looking at constraints from different linguistic levels (like coreference or discourse structuring). However, addressing these phenomena in a joint fashion is often not desired due to several reasons.

First of all, modeling system component interactions is computationally hard. Allowing pairwise interactions between the components raises the time complexity of an algorithm from linear to quadratic. Further increase in the number of interactions results in polynomial time. Unconstrained interaction, i.e. where all system components can influence each other, means an exponential number of component relations, modeling of which is intractable in practice.

Secondly, one needs to consider the need to maintain and debug an NLG system. Although joint modeling is a proliferate topic for research discussions and academic publishing, in practice system-building engineers often choose to use modular architectures, because they offer more transparency in their inner-working principles (Reiter and Mellish, 1993). Marr and Brenner (1976) provided another engineering argument in favor of modular architectures. The authors reasoned that any large computation should be split up into smaller independent parts. Otherwise, a small change in one place will have consequences in many other places. This, in turns, makes the process difficult to debug or improve, because a small chance to improve one part has to be accompanied by many simultaneous compensatory changes elsewhere.

Finally, a psychological argument in defense of the modular systems was provided by Reiter (1994). The author suggests that there is evidence that the human brain is modularized by nature. He cites studies of human patients with brain damage, and notes that such people tend to lose very specific, isolated language abilities, but do not suffer overall degradation that applies equally to all abilities. Brain damage sometimes results in losing the ability to organize utterances into coherent story, while not impacting the ability to produce syntactically correct utterances. Some patients exhibit the so-called "telegraphic speech": they utter a stream of content words, but leave out function words with grammatical significance.

The original modular pipeline systems were largely rule- or template-based (Inui et al., 1992; Reiter, 1994; Sripada et al., 2003b). Later, they adopted methods of data-driven machine learning. The current trend, at least in academic research, is to use end-to-end joint models which we briefly discuss next.

Joint models These approaches take a unified view of the NLG process and rely heavily on statistical inference that often cuts across task divisions, resulting in more integrated approaches to the NLG problem.

Data-driven systems often perform joint modeling implicitly, by learning the necessary information from input-output alignments. For example, Konstas and Lapata (2013) propose to induce a probabilistic context-free grammar (PCFG) from texts paired with meaning representations. Their system solves the tasks of document planning, content selection and surface realization jointly and at inference time incrementally generates textual descriptions of the input data. In a similar vein, Mairesse and Young (2014) describe a joint data-driven generation method that uses factored language models to search for the most likely sequence of semantic concepts and realization phrases.

Joint models do not have to be purely data-driven. But intersecting rules and templates that solve NLG subtasks is much harder than learning similar rules from corpora statistics automatically. An illustrative example is the development of *planning-based NLG systems* which view text generation as an execution of planned behaviour to achieve a communicative goal. This approach models NLG as a sequence of states, each of which corresponds to the change of physical or situated context, user's beliefs or actions, discourse history, etc. Deep reasoning about desires or intentions requires highly expressive formalisms which are not yet available, and researchers tried to approximate them with manually-crafted grammars (Hovy, 1988; Moore and Paris, 1993). These approaches started the line of work in planning-based NLG, but have lost their popularity, since it

became apparent that deterministic mapping between an action and its consequences is unrealistic, as new states are subject to uncertainty in operation conditions or user behaviour (Rieser and Lemon, 2009).

Another line of work tried to alleviate this limitation by using reinforcement learning (RL) techniques for modeling hierarchical relationships between sub-problems and performing their joint optimization (Lemon, 2008; Rieser and Lemon, 2011). This approach views NLG as a Markov decision process where states are associated with possible actions and each *(state, action)* pair is associated with a probability of moving from one state to another. Such architectures propose a less modular design, yet follow a general NLG subtask division and solve the subtasks jointly.

Other prominent approaches to joint data-driven NLG include framing the task as language modeling (Ratnaparkhi, 2000), using constrained optimization methods together with multi-step classification (Marciniak and Strube, 2005), inducing grammars which incorporate joint rules covering different subtasks (Belz, 2008; Wong and Mooney, 2007; Konstas and Lapata, 2012).

2.3.3 Deep Learning Approaches

The paramount of modern data-driven approaches are artificial neural networks. An exhaustive review of them goes beyond the scope of this thesis. We would like to refer the reader for further details to an excellent survey of various deep learning NLG approaches by Narayan and Gardent (2020). However, the importance of deep learning techniques for state-of-the-art NLP warrants a more detailed overview of the most prominent neural NLG approaches. Note that the architectures described below are general-purpose, and, broadly speaking, work for many text processing tasks, including, but not limited to, text generation.

Encoder-decoder architecture Most neural text generation approaches adopt the encoder-decoder approach pioneered by Sutskever et al. (2014) and Cho et al. (2014c). The general idea is to use an *encoder* neural network to encode the input data into a continuous representation, and then use a second network, a *decoder*, to incrementally generate an output sequence from it (Figure 2.9).

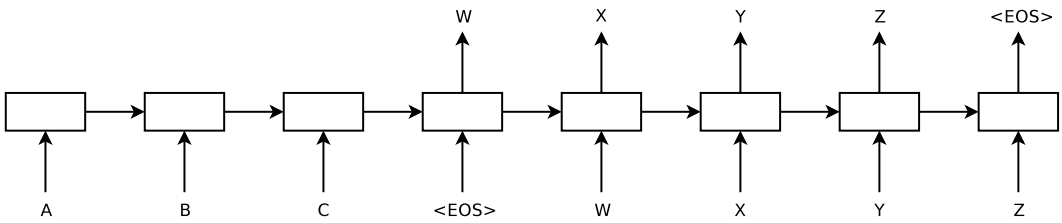


Figure 2.9: A schematic view of the encoder-decoder architecture. The model reads an input sequence *ABC* and produces *WXYZ* as the output sentence. Figure taken from Sutskever et al. (2014).

The architecture emerged to perform text-to-text generation in the context of machine translation, which is why this class of models is also called sequence-to-sequence (seq2seq). Such

models often use recurrent neural network (RNN) (Rumelhart et al., 1986) as an encoder and decoder, because RNN works well on arbitrarily long input sequences. An encoder processes an input sequence and compresses the information into a *context vector* of a fixed length, meant to be a summary of the meaning of the source sequence. A decoder operates as a language model conditioned on the encoded input sequence and the previously generated tokens.

To build input representations, early encoders used one-layer unidirectional RNNs to process the input sequence in a left-to-right fashion. Later approaches followed the work of Schuster and Paliwal (1997) and used bidirectional RNNs which permit creating input representations that take into account information from the whole input sequence, not just the preceding words. Stacking multiple RNN layers has also been found to improve system performance (Sutskever et al., 2014). While early work used vanilla RNN models, later research employed long short-term memory (LSTM) (Hochreiter and Schmidhuber, 1997) and gated recurrent unit (GRU) (Cho et al., 2014b) units which stabilize model training.

Such models quickly became a default method for text production, going beyond the machine translation community: Zhang and Lapata (2014) described a character-level RNN system to generate Chinese poetry; Rush et al. (2015) and Nallapati et al. (2016) pioneered the use of the encoder-decoder architecture in the task of abstractive text summarization; Hermann et al. (2015) and Hill et al. (2016) applied it to the task of machine reading comprehension, etc.

encoder-decoder models sidestep the necessity to align inputs and outputs in the training data, which makes them easy to use with any input data types. For this reason, and perhaps due to the absence of a unified way to treat structurally more complex inputs, seq2seq models have been used even for inputs that are not sequences. For example, previous work framed graph-to-text generation as a seq2seq task, where the input graph is "flattened" into a sequence by applying some heuristics, like depth-first graph traversal (Gardent et al., 2017; Castro Ferreira et al., 2017; Konstas et al., 2017). Figure 2.10 shows an example of how a constituency parse tree can be converted to a string of tokens using a depth-first traversal. Such input data can be further used for training a seq2seq constituency parser (Vinyals et al., 2015).

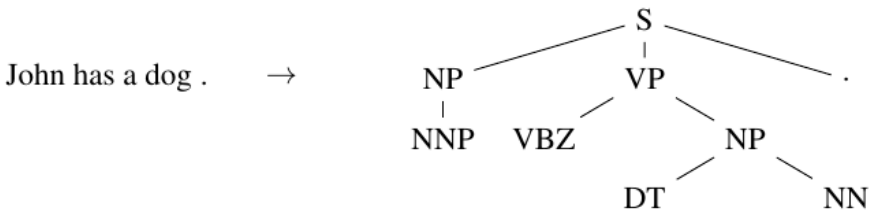


Figure 2.10: Example constituency parsing task and input linearization. Figure taken from Vinyals et al. (2015).

While models based on sequential encoders, like RNN, have shown good performance even when applied to complex non-sequential inputs, they fail to make the full use of the structure of the input data. More recent work has employed specialized encoders to generate texts from such

data: convolutional neural network (CNN) encoders for image-to-text generation (Xu et al., 2015b), graph convolutional network (GCN) (Kipf and Welling, 2017; Song et al., 2018; Marcheggiani and Perez-Beltrachini, 2018) for graph-to-text generation, etc.

Attention mechanism The encoding strategy of the vanilla encoder-decoder system is to compress the input into a fixed-length context vector, regardless of its length, which made it hard for the model to remember long input sequences. This has been shown to degrade model performance as the input length increases (Bahdanau et al., 2015).

The attention mechanism proposed by Bahdanau et al. (2015) addresses this problem: it creates a dynamic context vector by computing the similarity of the current decoder hidden state with each of the input tokens based on the corresponding encoder hidden states (Figure 2.11[3]).

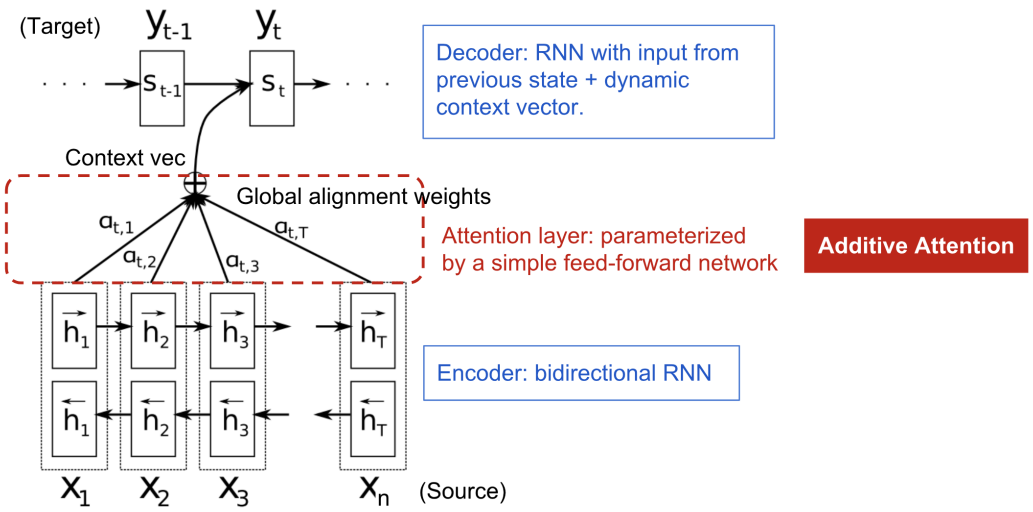


Figure 2.11: An encoder-decoder model with the attention mechanism of Bahdanau et al. (2015).

This technique also has been found to enable the models to learn alignments between different modalities, improving the performance of encoder-decoder systems across many NLP tasks (Mnih et al., 2014; Xu et al., 2015b; Chorowski et al., 2015). Importantly, this stimulated further research of other forms of attention mechanisms (Luong et al., 2015; Britz et al., 2017), leading to more efficient and complex models.

Transformer The most impactful work on improving the attention mechanism was that of Vaswani et al. (2017) who introduced the Transformer architecture. RNN-based systems are sequential by nature, because they factor computation along the token positions of the input and output sequences. This sequential nature hinders parallelization within training examples, which becomes critical at longer sequence lengths. The attention mechanism described above allows

[3]Image source: `https://lilianweng.github.io/lil-log/2018/06/24/attention-attention.html`

modeling of dependencies without regard to their distance in the input or output sequences, but its usage was tied to RNN-based approaches.

The Transformer removes the recurrent connections of RNN, and instead attends to the entire input sequence simultaneously. The attention mechanism employed by the Transformer relates different positions of a single sequence in order to compute a representation of the sequence, which is why it is also called *self-attention* (Parikh et al., 2016; Lin et al., 2017).

The Transformer follows the overall encoder-decoder architecture, and uses stacked self-attention and point-wise fully connected layers for both the encoder and decoder (Figure 2.12).

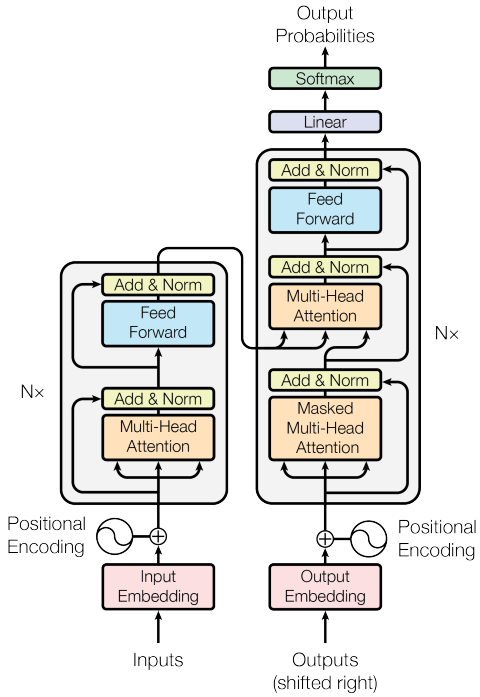


Figure 2.12: The Transformer architecture. $N\times$ denotes a stack of N layers. Figure taken from Vaswani et al. (2017).

There are other novel architectural choices that the authors of the Transformer proposed in order to improve the efficiency of the model. For example, instead of computing a single attention function, the Transformer uses several parallel attention layers, or heads (*multi-head attention*). These attention heads process the sequence in parallel; their outputs are then concatenated and projected to a lower-dimensional space. Multi-head attention allows the model to jointly attend to information from different representation subspaces at different positions. Since the model contains no recurrence and no convolution, in order for the model to make use of the order of the sequence, the authors proposed to use *positional encodings* which inject some information about the relative or absolute position of the tokens in the sequence.

Compared to RNN-based systems, The Transformer architecture has been shown to be both faster and more accurate across a range of tasks, and quickly became the main encoder-decoder

architecture in NLP. The popularity of the Transformer in NLG is linked to its usage for pretraining and fine-tuning large generative models, which we briefly describe below.

GPT and BERT When trained from scratch for a specific task, deep learning models require large amounts of data and may need significant amount of time to train. Transfer learning in the form of general-purpose generative pretraining of a language model, followed by task-specific fine-tuning has become a common way of knowledge transfer between neural NLP models (Dai and Le, 2015; Peters et al., 2018; Howard and Ruder, 2018). The idea is to pretrain a language model on a large general-domain corpus (like Wikipedia), and then continue training this pretrained model on a smaller, in-domain, task-specific dataset (fine-tuning). This pretraining causes the model to develop general-purpose abilities andknowledge that can then be transferred to downstream tasks (Raffel et al., 2020).

The Transformer architecture described above has found its use as a base for transfer learning models in the OpenAI Generative Pretrained Transformer (GPT), a left-to-right, decoder-only multi-layer text generation model (Radford et al., 2018). The authors showed how a generative language model is able to acquire world knowledge and process long-range dependencies by pretraining on a diverse corpus with long stretches of contiguous text. Unlike previous approaches which required task-specific adjustments to the system architecture for fine-tuning purposes (Peters et al., 2018), GPT offered a very general NLP framework which can be easily adapted to many NLP tasks (Figure 2.13).

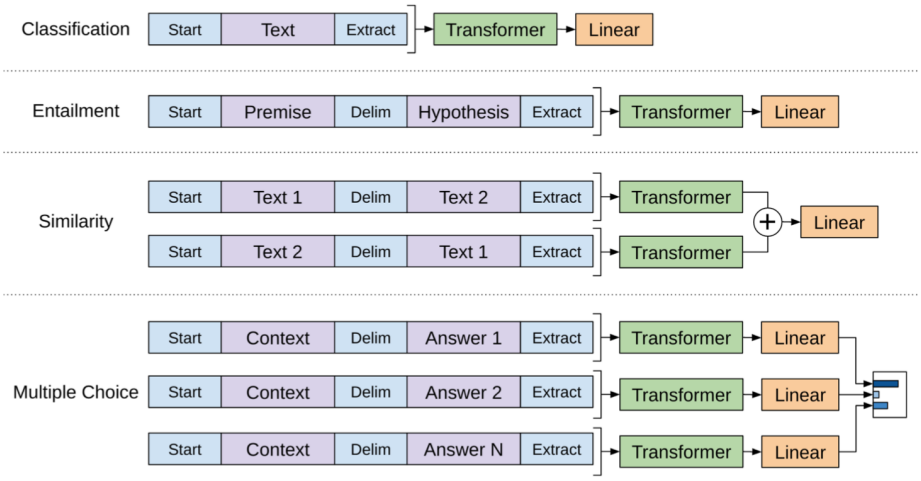


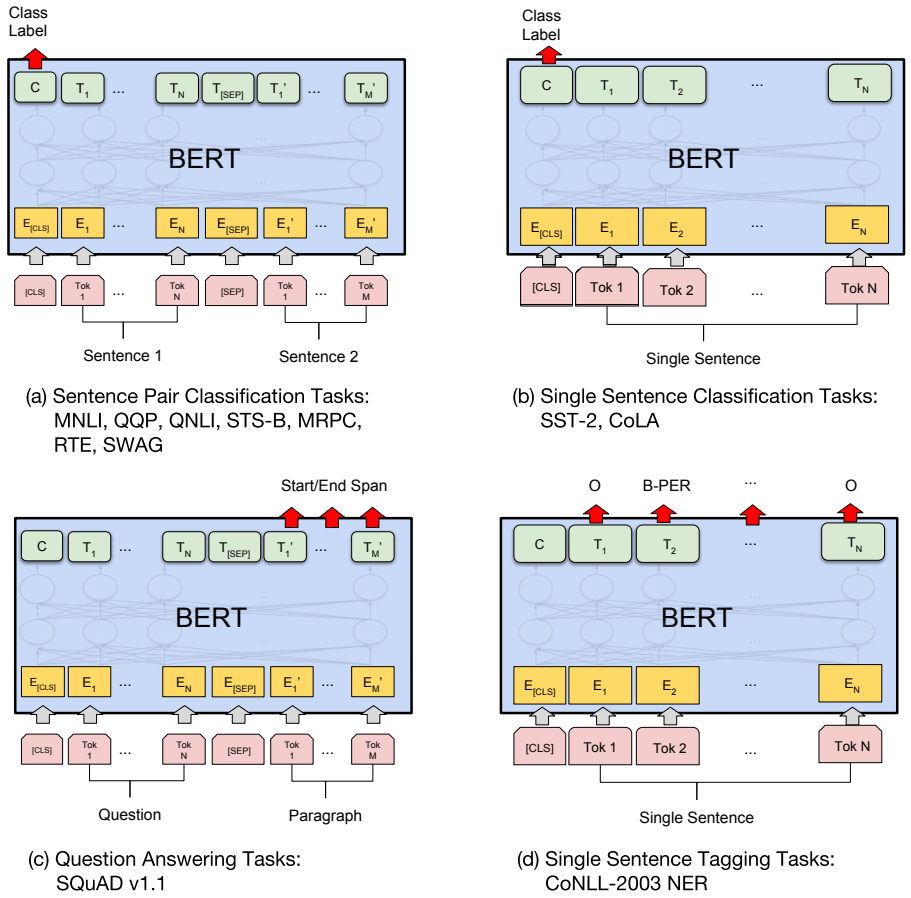
Figure 2.13: GPT fine-tuning: input transformations for four different NLP tasks. Figure taken from Radford et al. (2018).

As we mentioned, GPT is uni-directional, i.e. it processes inputs in a left-to-right fashion. Similar to how uni-directional RNN was extended by its bi-directional variant, a bi-directional version of the Transformer appeared as an extension of GPT. Bidirectional Encoder Representations from Transformers (BERT) is a language representation model designed to pretrain deep bidirectional representations by jointly conditioning on both left and right context (Devlin et al., 2019).

BERT achieves that by using a *masked language modeling* (MLM) training objective: some of the tokens are randomly masked from the input, and the objective is to predict the masked word based on its context. Unlike left-to-right language model pretraining, the MLM objective allows the representation to fuse the left and the right context, which allows one to pretrain a deep bidirectional Transformer.

Many downstream NLP tasks, such as question answering and natural language inference, are based on understanding the relationship between two text snippets, which is not directly captured by language modeling. In order to train a model that understands sentence relationships, the authors proposed to use an additional *next sentence prediction* (NSP) objective which jointly pretrains text-pair representations.

Similar to GPT, the BERT architecture is easily adapted for fine-tuning models in many NLP tasks (Figure 2.14).



(a) Sentence Pair Classification Tasks: MNLI, QQP, QNLI, STS-B, MRPC, RTE, SWAG

(b) Single Sentence Classification Tasks: SST-2, CoLA

(c) Question Answering Tasks: SQuAD v1.1

(d) Single Sentence Tagging Tasks: CoNLL-2003 NER

Figure 2.14: BERT fine-tuning: input transformations for several NLP tasks. In the figure, E represents the input embedding, T_i represents the contextual representation of token i, CLS is the special symbol for classification output, and SEP is the special symbol to separate non-consecutive token sequences. Figure taken from Devlin et al. (2019).

BERT, GPT and their derivatives, like BART (Lewis et al., 2020) or T5 (Raffel et al., 2020), DIALOGPT (Zhang et al., 2020b) constitute the current state-of-the-art in NLP, as measured by common NLP benchmarks, like The Stanford Question Answering Dataset (Rajpurkar et al., 2016), the General Language Understanding Evaluation (Wang et al., 2018), or Reading Comprehension from Examinations (Lai et al., 2017). Considerable efforts are being applied to make BERT-like models more efficient by reducing the number of parameters (Lan et al., 2020; Sanh et al., 2019; Chen et al., 2020a) and improving the training procedure (Yang et al., 2019; Liu et al., 2019c; Clark et al., 2020). GPT-style models have been evaluated in experiments testing the limitations of scaling to larger model and data sizes. The OpenAI GPT model was followed by GPT-2 (Radford et al., 2019), a model with 10 times more parameters than the original GPT model. GPT-3 (Brown et al., 2020) has the same architecture as GPT-2, but is an order of magnitude larger than GPT-2. The latter two have been shown to exhibit non-trivial zero-shot transfer capabilities: GPT-3 achieves state-of-the-art results on the language modeling, unsupervised machine translation, question-answering and cloze tasks in a zero-shot transfer setting even without any task-specific fine-tuning.

2.3.4 Summary

To conclude, the intersection of the two described typologies produces four system type variants (Table 2.1).

Property	JointRule	ModularRule	JointData	ModularData
White-box	+	+	−	+/−
Controllable	+	+	−	+/−
Task-agnostic	−	−	+	+
Domain-agnostic	−	−	+	+
Scalable	−	+/−	+	+
Accurate	+	+	data-dependent	data-dependent
High output variation	−	−	data-dependent	data-dependent

Table 2.1: A high-level comparison of joint/modular and rule-based/data-driven systems in NLG.

Rule-based systems (both joint and modular) offer a holistic, principled NLG approach, but can only be developed in restricted domains with limited variation of linguistic phenomena.

While rule-based systems allow more control over system design, the burden of rule and template definition, coverage issues and difficulties with porting them to other problems make them less attractive than their data-driven counterparts. The latter not only scale better, but also lift the requirement of possessing domain knowledge from system engineers; the domain-specific aspect becomes a property of the training data.

Pipeline data-driven approaches are typically represented as sequences of classification decisions. They maintain a clean separation between tasks, but suffer from the error propagation problems, whereby classification errors impact decisions further downstream.

A potential solution to this issue is offered by data-driven joint approaches which view generation as an optimization problem, where the best combination of decisions is sought in an exponentially large space of possible combinations. This can be done via constrained optimization methods or reinforcement learning techniques.

The dominant joint data-driven approach is represented by deep neural networks. Deep learning development produced a rich and diverse family of approaches, but in NLG tasks they usually adopt the encoder-decoder architecture. NLG (and NLP in general) has shifted from learning task-specific representations and designing task-specific architectures to using task-agnostic pretraining and task-agnostic architectures, like RNN and, more recently, the Transformer. Further improvements can be gauged by increasing the training data and computation power, by making the models more efficient, or by improving the training procedure. The biggest disadvantage of deep learning approaches is their black-box nature: unlike rule-based and pipeline data-driven approaches, deep learning models are hard to analyze, due to their complexity (Hale et al., 2018; Futrell et al., 2019; Tenney et al., 2019; Mrini et al., 2020).

Now that the reader is familiarized with the NLG task, data and common techniques, we would like to move on to the question of system evaluation. In the next section we discuss the established evaluation protocols in NLG research. We will describe types of evaluation that researchers often use to assess the performance of the NLG systems, and the most common metrics that are employed for system development and automatic quality assessment.

2.4 Evaluation

NLG system evaluation is challenging for several reasons. First of all, the variety of input types and formats makes it hard to use benchmarks. Even when the output specification is clear, different systems assume different levels of pre- or post-processing which often involves the usage of standalone linguistic analyzers, giving some systems an advantage over the others. Shared tasks and workshops organized by the NLP community are of great help in this regard, because common rules of participation offer a fair ground for comparison of the submitted system outputs. Some shared task organizers acknowledge the importance of pushing the scientific frontiers further and offer unconstrained tracks which encourage the participants to use any training data and additional NLP tools to solve the proposed tasks. Such competitions help to understand the limitations of modern approaches and often serve as a discussion fora for the community.

Another difficulty is posed by the limitations of modern experiment design. Many data-driven systems are biased towards certain types of evaluation setup. For example, unregularized deep learning approaches perform poorly under a low-resource scenario, but often beat other techniques when the training data is abundant (Koehn and Knowles, 2017; Ma et al., 2019b; Chen et al., 2020b). Some methods can operate on very noisy inputs, while others seem to be sensitive to data artefacts and perform much better on clean data. This makes it hard to assess the actual potential of NLG systems in real-world applications, and ultimately calls for the development of principled evaluation frameworks which could give an objective assessment of the available techniques and offer a holistic explanation of the results.

2.4.1 Typology

2.4.1.1 Intrinsic and Extrinsic

Prior work proposed a few prominent approaches to categorization of evaluation methods for NLG. The most common methodological distinction is that between intrinsic and extrinsic evaluation methods (Jones and Galliers, 1995). The basis for this distinction can be defined as *reference to system-agnostic properties of the evaluation setup*. Intrinsic evaluation assesses an approach without reference to its effectiveness in relation to the users. Extrinsic methods measure system performance with respect to achieving a desired goal. They estimate the system's utility, impact or effectiveness for a task or application.

A related categorization was proposed by Reiter and Belz (2009) who distinguished between task-based evaluation, human ratings, and corpus comparison using automatic metrics. The authors do not provide a basis for this classification, but we can probably define it as *the source of the gold standard, the reference signal*. This categorization overlaps with the first one: task-based evaluation is extrinsic, while metric- and human-based are intrinsic. For ease of exposition, we intersect them and briefly describe below (Table 2.2).

Property	IntrinsicHuman	IntrinsicMetric	ExtrinsicTask
Fast	+/−	+	−
Cheap	+/−	+	−
Expert-agnostic	+/−	+	−
Reliable	+/−	−	+
High utility	+/−	−	+

Table 2.2: A high-level comparison of evaluation types in NLG.

Intrinsic, human-based In human experiments for intrinsic evaluation, the subjects are asked to read and rate output texts according to some criteria. This evaluation focuses on getting direct human feedback on certain aspects of text quality. The assessors are usually asked to rate generated texts on a 5-point rating scale, or state their preference between system outputs. Examples of the evaluation criteria which are commonly measured using human ratings include overall quality, coherence, content, organization, writing style (Lester and Porter, 1997).

Evaluation based on human ratings is typically quicker and cheaper than task-based evaluation. It also does not require as much support from the experts. However, human rankings tend to be less reliable than task-based evaluation, and depend on many factors of human experiment design: number of participants (Rus et al., 2011), type of the scales employed (Belz and Kow, 2010, 2011), expected level of expertise (Goodman et al., 2013; Mason and Suri, 2012), etc.

Intrinsic, metric-based Intrinsic evaluation can also be done automatically, by computing the scores of evaluation metrics and ranking the systems according to the obtained scores. The scores are usually computed as a result of comparison between system outputs and human references,

although there exist metrics that are reference-free. There is a great variety of metrics that are commonly used in the NLG community; Section 2.4.2 provides a more in-depth description of the ones that will be mentioned in this thesis.

Metric-based evaluation is more popular among some NLG subtasks than the others. For example, it is widely used when evaluating surface realizers, because corpus-based techniques are suited for measuring grammatical coverage (Reiter and Belz, 2009). In addition, there is little variation in terms of content determination, microplanning and lexical choice, which is why few references are needed to adequately reflect the solution space.

The main advantages of metric-based evaluations are their cost-efficiency and speed. They also do not rely on domain experts as much as, for example, task-based evaluation. Consequently, such experiments are also more repeatable and reproducible.

The biggest disadvantage of metric-based evaluation is its reliance on various metric assumptions that do not always hold in a particular setup. For example, some metrics assume that corpus texts are of good quality, which is not always the case (Sripada et al., 2003a). Some common metrics fail to adequately assess structural aspects of the output (Scott and Moore, 2007). It has also been shown that metric optimization often leads to "gaming" the task, i.e. achieving a high metric score, while ignoring the main objective of the task — solving the problem (Turian et al., 2003; Sun et al., 2019).

Another disadvantage of metric-based evaluation is the need to have a corpus of human-written reference texts (possibly with multiple references). Sometimes gold-standard texts are hard to acquire. For example, Reiter and Belz (2009) reported that in the medical domain it can take an experienced clinician several hours to write a corpus text from the raw data. Creating a corpus of 100 reference texts could require 2–3 months of effort by an expert doctor. This could be difficult, unless a very strong case could be made for the utility of the evaluation.

Extrinsic, task-based Task-based evaluation measures the impact of the generated texts on the end users, the system *effectiveness*. The notion of effectiveness is task-dependent; below are sample definitions from prior NLG work:

- number of mistakes while carrying out a task (Young, 1999)
- learning gain from adding an NLG component to an intelligent tutoring system (Di Eugenio et al., 2002, 2005; Boyer et al., 2009)
- persuasion and behaviour change (Reiter et al., 2003; Carenini and Moore, 2006),
- purchasing decision after presentation of arguments for and against options on the housing market based on a user model (Carenini and Moore, 2006),
- decision support in a medical setting following the generation of patient reports (Portet et al., 2009),
- enhancing linguistic interaction among users with complex communication needs via generation of personal narratives (Black et al., 2010),
- engagement with ecological issues after reading blogs about migrating birds (Siddharthan et al., 2012),

The main advantage of task-based evaluations is their utility in assessing the usefulness of a scientific method in a real-life application. They are considered to be very persuasive when it comes to convincing other researchers or prospective users. However, task-based evaluation is very costly and time-consuming. In addition, careless participants may disrupt the experiment and yield the results of the study useless. Also, task-specific performance may not correlate with other related tasks (Carter, 1996).

How to measure the effectiveness of an approach is also an important decision to be made when performing task-based evaluation. It can be done by asking the participants to fill out questionnaires after task-completion, measured automatically during the study, or collected after system deployment in the real world. Choosing one option over the other may have long-lasting consequences on the results of the study, which is one of the main drawbacks of extrinsic methods in general.

2.4.1.2 Other Evaluation Types

Belz (2009) mentions a division of evaluations into user-oriented and developer-oriented, based on the *evaluation purpose*. The former is based on a set of user requirements, like available resources, acceptable processing time, maintenance costs, etc. They assess how well different technological alternatives fulfill them. Developer-oriented evaluations focus on system functionality.

Popescu-Belis (2007) add another dimension to the evaluation categorization: *the role of language in the input and output of an NLG system*. The authors divide all systems into the following groups:

- analytic systems which use linguistic units as input,
- generative systems which use language as output,
- analytic generative systems that use language both as input and as output,
- interactive analytic generative systems which interact with human users to produce a result.

According to the authors, input and output specification dictates the evaluation criteria and measures to assess NLG systems. For example, analytic systems can be evaluated using distance-based comparison between a reference and a system output. However, the evaluation of generative systems is complicated by the fact that the range of possible outputs is not restricted, due to human language variability. Also, it is close to impossible to define one shared task evaluation for all NLG systems of the generative type, since different systems have different inputs and outputs. This implies that one cannot use distance-based metrics for system evaluation and has to resort to other methods (human ratings or task-based evaluation).

Finally, some researchers propose to distinguish between evaluations of NLG system components, end-to-end systems and embedded NLG components (Hastie and Belz, 2014). While it is rather hard to put different systems on equal grounds in this categorization framework, the principles described by the authors and the analysis of various community efforts offer valuable guidance to those interested in NLG evaluation from an engineering perspective.

2.4.2 Common Metrics

Human experiments and task-based evaluations are sometimes used to validate the findings of intrinsic metric-based evaluation. In this section we briefly describe the automatic metrics most commonly used in the NLG literature.

2.4.2.1 N-gram Matching

N-gram matching metrics compute an overlap of contiguous sequences of n tokens (n-grams) between system outputs and references.

BLEU Proposed by Papineni et al. (2002), BLEU is a precision metric that calculates the score of a text by measuring the number of n-grams of varying length of the system output that occur within a set of references. Each n-gram in the reference can be matched at most once. The number of exact matches is accumulated for all reference-candidate pairs in the corpus and divided by the total number of n-grams in all candidate sentences. A special brevity penalty is used to discourage very short candidates.

The metric originated in the machine translation community to evaluate the quality of system translations. BLEU scores range from 0 to 1, where 1 is the highest one which can only be achieved by a generated text if all its substrings can be found in one of the reference texts; higher-order n-gram overlap is meant to capture fluency considerations.

There is a sentence-level smoothed variant of BLEU (SentBLEU), but to the best of our knowledge, it is rarely used beyond the machine translation community (Koehn et al., 2007).

NIST NIST evaluation score (Doddington, 2002) is an adaptation of BLEU that gives more weight to less frequent n-grams which are assumed to be more informative. In machine translation, this metric has shown a higher correlation with human judgements of adequacy, i.e. how well the meaning conveyed by the reference translation is also conveyed by the evaluated segment (Doddington, 2002).

ROUGE ROUGE is a family of recall-oriented metrics used predominantly in document summarization. It includes a vanilla n-gram overlap variant, as well as modifications which compute non-contiguous n-grams and longest common subsequences (Lin and Hovy, 2003).

The simplest metric variant, ROUGE-N, compares a text to a set of references and computes the highest proportion of n-grams of length N that are matched by the generated text and any of the references. The score is averaged across leave-one-out subsets of the set of reference texts. A text can get a score of 1 only if there is only one reference which is identical to the text candidate.

Another variant of the metric, ROUGE-SUN considers the so-called *skip-bigrams* in the generated text and reference texts. A skip-bigram is two words which are not necessarily adjacent, but may be separated by up to N intermediate words. ROUGE-2 and ROUGE-SU4 are the main metrics used in the summarization community.

METEOR METEOR is an automatic metric from the machine translation community, which evaluates a translation by computing a score based on explicit word-to-word matches between the translation and a given reference translation (Banerjee and Lavie, 2005). If more than one reference translation is available, the translation is scored against each reference independently, and the best scoring pair is used.

METEOR has a number of advantages over other n-gram metrics. First of all, it goes beyond simple n-gram matching and accounts for synonyms in the outputs. Secondly, its score includes computation of both precision and recall. Compared to purely recall-oriented (ROUGE) or precision-oriented (BLEU, NIST) metrics, it is more robust, because a system can inflate its recall score by outputting every word from the vocabulary, and achieve perfect precision by just outputting one most common token, like "the" (Turian et al., 2003). Finally, METEOR by design is a segment-level metric, which in some cases makes its scores more reliable than corpus-level ones, like BLEU.

In machine translation, METEOR has been shown to outperform other n-gram metrics (Graham et al., 2015). However, one big disadvantage of it is that it is language-specific, since its computation relies on using WordNet for finding the synonyms.[4] METEOR also relies on external tools (a stemmer, a synonym lexicon, and a paraphrase table), which makes its computation much slower, compared to other metrics.

GTM General Text Matcher (GTM) (also known as F-measure) computes an n-gram overlap as a harmonic mean of precision and recall with greater weight for contiguous matching spans (Turian et al., 2003). Despite the encouraging results presented in the original paper, the metric does not seem to be very popular in the NLP community.

CIDEr Consensus-based Image Description Evaluation (CIDEr) score for n-grams of length n is computed using the average cosine similarity between the candidate sentence and the reference sentences, which accounts for both precision and recall. The n-grams are stemmed and weighted using term frequency–inverse document frequency (tf-idf) (Vedantam et al., 2015). CIDEr originated in the image captioning domain and is predominantly used for evaluating short textual snippets.

Word-Mover's Distance Word-Mover's Distance (WMD) measures the dissimilarity between two texts as the minimum amount of distance that the embedded words of one text need to "travel" to reach the embedded words of another text (Kusner et al., 2015). This metric was inspired by the success of word embedding representations learned from local co-occurrence statistics in sentences (Mikolov et al., 2013; Luong et al., 2013; Pennington et al., 2014, *inter alia)*.

This is a comparatively new metric, but it has already been shown to correlate well with human judgements of image caption quality (Kilickaya et al., 2017).

[4]METEOR offers full support for five and partial support for eleven languages (Zhang et al., 2020a).

2.4.2.2 String Matching

Edit Distance String edit distance is measured as a number of insertions, deletions, substitutions and transposition operations which are required to transform one string into another (Levenshtein, 1966).

The scores are often normalized to range from 0 to 1, where the latter means a perfect match between two strings. When multiple references are used, the score for a generated text is the average of its scores against the reference texts. The edit distance score for a set of generated texts is the average of scores for the individual texts.

TER Translation Error Rate (TER) measures the amount of editing that a human would have to perform to change a system output so it exactly matches a reference translation (Snover et al., 2006).

TER has been shown to correlate well with the human evaluation of the translation adequacy (Snover et al., 2006; Espinosa et al., 2010). The original version has several flaws, however. First of all, it only considers exact matches when measuring the similarity of the hypothesis and the reference. Secondly, it can only compute the measure of similarity against a single reference. TER-Plus (or TERP) extends the TER metric beyond the limitation of exact matches through the addition of three new types of edit operations: stem matches, synonym matches and phrase substitutions (G. Snover et al., 2009).

2.4.2.3 Vector-space Metrics

Following the development of deep learning models, NLG evaluation community started exploring the potential of pretrained language models for automatic evaluation of texts using *vector-based metrics*. Contextual embeddings generated from such models have been found to be effective for paraphrase detection and capturing distant dependencies and ordering (Devlin et al., 2019).

BERTScore BERTScore (Zhang et al., 2020a) computes the similarity of two sentences as a sum of cosine similarities between their tokens' contextual embeddings generated by a pretrained BERT model. BERTScore was conceived to address such shortcomings of n-gram overlap metrics, as failure to robustly match paraphrases (Banerjee and Lavie, 2005) and failure to capture distant dependencies and penalize semantically-critical ordering changes (Isozaki et al., 2010). BERTScore has been shown to outperform existent metrics in the tasks of machine translation and image captioning (Zhang et al., 2020a).

BLEURT BLEURT (Sellam et al., 2020) is another learned evaluation metric based on BERT, which has been shown to perform well in the machine translation and data-to-text generation tasks. Unlike BERTScore which directly uses outputs of a pretrained BERT model, BLEURT fine-tunes the pretrained BERT model in two steps. First, a large number of synthetic reference-candidate pairs is generated, and BERT is fine-tuned on this data with several lexical- and semantic-level supervision signals under a multi-task loss. Then the model is additionally fine-tuned on human references.

The metrics we described above use some form of a reference as gold standard. This puts a lot of responsibility on the reference collection step, which sometimes is not paid due attention to. Poor-quality or insufficient references can easily sabotage the results of the correlation study. However, this aspect seems to be under-explored in the research literature. To the best of our knowledge, most correlation studies attempt to prove metric correlation, as opposed to finding why established metric might not correlate. Notable exception works include (Reiter and Sripada, 2002; Reiter and Belz, 2009; Novikova et al., 2017b) which advocate for the development of new NLG metrics. Vector-based metrics based on large-scale pretrained language models offer an opportunity to side-step this issue: in essence, these metrics are reference-less and, therefore, can be used even in cases where references are noisy.

2.4.3 Human Judgments

As mentioned in Section 2.4.1, intrinsic human-based evaluation is a common way to assess NLG systems. Human judgments in NLG are usually elicited via a direct assessment of system outputs, or their comparisons.

Direct assessment Discrete scales are considered to be the dominant method: a human assessor is asked to rate the system output on a n-point rating scale, thus directly estimating the quality of the system output (Figure 2.15). The most common value of n is 2, 3, 5, or 7 (Van der Lee et al., 2019).

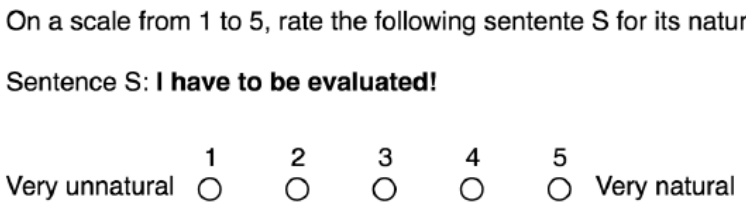
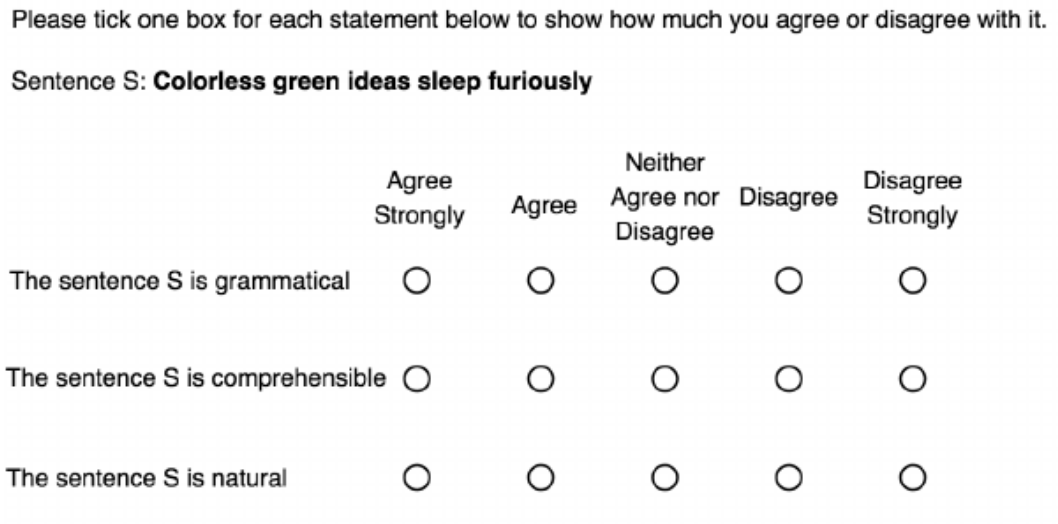


Figure 2.15: An example of a numerical rating scale. Figure taken from Amidei et al. (2019).

Sometimes, a Likert scale (an aggregate of graphic rating scales) is used in order to collectively capture the phenomenon under analysis. Figure 2.16 shows an example of a Likert scale: the quality of a sentence under consideration is being assessed along three axes (grammaticality, comprehensibility and naturalness). The collective assessment is produced by summing or averaging the individual measurements to produce a total qualitative score.

Some works argue that a continuous scale gives human raters more control over the direct assessment process and releases them from the distress experienced when the judgment falls between the judgment points (Belz and Kow, 2011). Continuous scales usually have a slider that allows the participants of the study to choose any score value, typically between 0 and 100. An example of a continuous scale is shown in Figure 2.17.


Please tick one box for each statement below to show how much you agree or disagree with it.

Sentence S: **Colorless green ideas sleep furiously**

	Agree Strongly	Agree	Neither Agree nor Disagree	Disagree	Disagree Strongly
The sentence S is grammatical	○	○	○	○	○
The sentence S is comprehensible	○	○	○	○	○
The sentence S is natural	○	○	○	○	○

Figure 2.16: An example of a Likert scale. Figure taken from Amidei et al. (2019).

Output comparison As Van der Lee et al. (2019) point out, Likert scale is the most popular elicitation method in the NLG community. Perhaps, the main reason for this is the simplicity of the task design. However, the scale is very rigid in a sense that it does not give human assessors more fine-grained control over evaluation. It also forces a rater to commit to a scoring decision, which in practice is not that easy to make; oftentimes, expressing a preference judgment is easier.

More recent studies show that preference-based evaluation produces more reliable and consistent results. This paradigm is represented by ranking methods, whereby system outputs are compared instead of being scored. There is growing evidence that such methods produce more consistent and more discriminative human ratings than direct assessment methods (Callison-Burch et al., 2007; Yannakakis and Hallam, 2011; Novikova et al., 2018). Two main disadvantages of ranking-based methods are a comparatively large number of comparisons needed to produce reliable estimates, and the need for a method that would aggregate them to produce system ranking (Callison-Burch et al., 2007). Prior work in NLG has explored the utility of algorithms developed for modeling the relative skills of players in ongoing competitions. A notable example of such algorithms is TrueSkill™, an adaptive model of competitions, originally developed for the online gaming community (Herbrich et al., 2006). The algorithm assumes that each player's skill level follows a probability distribution with a player's mean performance and the system's uncertainty about its current estimate of this performance as parameters. These parameters are updated after each "game" in proportion to how surprising the outcome is. This method has been adapted to a number of research areas, including NLG where it has been shown to reliably induce system rankings, even when few system comparisons are available (Sakaguchi et al., 2014; Novikova et al., 2018).

Table 2.3 provides a summary of direct assessment vs. output comparison along several dimensions.

Q1: Grammaticality The summary should have no datelines, system-internal formatting, capitalization errors or obviously ungrammatical sentences (e.g., fragments, missing components) that make the text difficult to read.

1. Very Poor
2. Poor
3. Barely Acceptable
4. Good
5. Very Good

(a) Discrete Scale

Q1: Grammaticality The summary should have no datelines, system-internal formatting, capitalization errors or obviously ungrammatical sentences (e.g., fragments, missing components) that make the text difficult to read.

(b) Continuous Scale

Figure 2.17: An example of a discrete and continuous rating scales. Figures taken from Belz and Kow (2011).

2.5 Chapter Summary

The purpose of this chapter was to provide sufficient background information for the reader to get an understanding of the NLG task.

We first analyzed the structure of the language generation process, and used examples from prior work to show that language generation activity consists of several distinct steps. We found that the related work exhibits some variation in the exact composition and naming convention, but overall shares the idea that NLG is a complex process involving several stages which can be computationally modeled, in isolation or jointly.

We further described the input-output specification of the task, with examples to showcase the variety of data types that NLG systems use:non-linguistic data, structured text representations, and raw texts.

Each input data type has its limitations and requires the right approach to be used with. We provided a high-level overview of the dominant methods used in the NLG field. They range from domain-specific rule-based and template systems to end-to-end data-driven models, and are characterized by different trade-offs one needs to consider when approaching the task at hand.

Finally, we described how NLG systems are typically evaluated. Extrinsic task-based evaluation is the most solid and convincing option, since it directly assesses the utility of the system for a specific task and target audience. However, it is also the most resource-consuming and, therefore, least reproducible one. Intrinsic human-based evaluation is a cheaper alternative, but because it uses human experiments as a proxy for the real task, it needs to be carefully designed and

Property	DirectAssessment	RankingComparison
Controlled assessment	−	+
Consistent	+/−	+
Reliable	+/−	+
Simple to design	+	+/−
Annotation difficulty	+/−	+
# points to compare	+	−

Table 2.3: A high-level comparison of the two main strategies to elicit human judgements in the NLG experiments: direct assessment and output comparison. The former includes rating and Likert scales, the latter refers to ranking methods.

monitored to ensure the reliability of the results. Finally, the cheapest option is intrinsic metric-based evaluation: it can be run automatically and does not require human subjects, which ensures it can be easily reproduced. The biggest drawback of metric-based evaluation is the uncertainty about the correlation between the metric values and human judgements of system output quality.

We further briefly described the most popular evaluation metrics used in the NLG community. We provided some evidence from prior work about whether the mentioned metrics have been found to correlate well with human judgements. The main conclusion is that the evidence is often conflicting, which makes metric choice a hard problem.

Finally, we briefly described several most common methods of eliciting human judgements when performing intrinsic human evaluation. It seems that we are currently witnessing a paradigm shift: the community is moving from discrete rating and Likert scales to continuous scales and preference-based evaluation.

CHAPTER 3
Sanity Polygon Framework

In the previous chapter we described the NLG task specification, the dominant approaches to automatic text production and the existent evaluation methodology.

In this chapter, we will introduce the central problem addressed in the thesis — the absence of a principled methodological approach to NLG research. First, based on the review of related work described in the previous chapter, we motivate the need to develop such an approach. We further examine several critical issues posed as challenges one faces when developing an NLG system. We describe our recommendations and formalize them as a checklist that could be used as approach development guidelines. In the subsequent chapters we describe several studies that demonstrate how it can be used in the NLG experiments.

3.1 Motivation

The related work overview which we presented in the previous chapter shows that NLG research is gaining its momentum. Several decades ago NLG was limited to hand-crafted rule-based systems mainly assisting professionals in very specific applications. Modern NLG research has crossed the application and domain boundaries and now forms a universal text generation paradigm.

Despite the growing attention to the field, NLG research has a lot of gaps that make it hard to develop robust NLG systems. The existent NLG surveys either provide a high-level overview of the developed techniques (Gatt and Krahmer, 2018), or target a specific problem, like NLG evaluation issues (Gkatzia and Mahamood, 2015; Amidei et al., 2018; Van der Lee et al., 2019). We acknowledge the importance of such research and ourselves build our work upon it, but also argue that one of the obstacles impeding further NLG development could be the lack of principled, systematic methodology to approach NLG studies:

- The community has developed a deep understanding of what NLG is, which subtasks it has, but the architectural differences and different input–output specifications of various systems make it hard to compare them.

- There are gaps in understanding which approaches are applicable in which task setup. NLG literature has explored a great deal of techniques, but in practice researchers try out several recent systems to figure out which performs best. We argue that this practice can be improved by a deeper analysis of the task at hand and following some methodological principles.
- NLG evaluation is still a "black-box": research papers discussing the correlation of various metrics keep appearing in conference proceedings, but there is hardly any consensus on what works and what does not. This ultimately causes confusion and leads to the usage of a large battery of non-discriminative metrics, which gives rise to contradicting conclusions.

We believe that there is an acute need for a more principled approach to performing NLG studies, and aim at developing a methodology which could help one face the chronic under-specification of the NLG task. The importance of this problem is confirmed by the growing number of collaborative efforts aimed at improving the methodology of NLG research. For example, a few of the most recent international venues devoted to the problems of NLG evaluation include the following:

- Evaluation and Comparison of NLP Systems (Eger et al., 2020): aimed at the design of adequate metrics, evaluation methodology and creating correct evaluation data.
- Workshop on Evaluating NLG Evaluation:[5] aimed at developing methodology and linguistic aspects of evaluation, rather than the proposal of new automatic metrics.
- Shared Task on Evaluating Accuracy (Reiter and Thomson, 2020): a shared task on methodologies and algorithms for evaluating the accuracy of generated texts.

However, system evaluation is only one part of the NLG process. In this chapter we present a conceptual framework which we developed for analyzing and planning NLG experiments. It offers a holistic view on NLG,[6] and is based on several principles governing the relation between the major components of any NLG study: the task, training data, evaluation criteria, automatic metrics for system development and human experiment study design. We describe each of the principles and use the identified principles in a series of studies to showcase our ideas.

3.2 Framework Overview

Research questions in NLP always have an underlying real-world problem:

- translating a document from one language to another,
- summarizing a document collection,
- finding an answer to a question,
- transforming the data representation from one modality into another (e.g. image → text),

[5] `https://evalnlg-workshop.github.io/`

[6] By "holistic" we mean that NLG should be approached as a coherent whole, and argue that its components are best analyzed in relation to one another.

- responding to a user query.

Based on the problem, a researcher defines a **(1) task** and first decides what the inputs and outputs should look like, and what the system requirements are. Depending on the task specifications and requirements, they need to establish the resources available at his or her disposal. One of the most important ones is **(2) data** needed to develop and evaluate the proposed solutions. Data-driven approaches use machine learning algorithms to find statistical regularities in the available data, and usually express them in a form of a parameterized function, where the parameters (also called weights) are learned during training. Many problems in NLP trace back to data issues; this dependence is often described as the *garbage in, garbage out* empirical law which means that noisy input data usually result in nonsense outputs.[7]

Evaluation specifications govern the decision of which approach one should choose. Ultimately, a researcher is interested in a positive evaluation of the outputs produced by an envisioned system. This means the researcher needs to understand which facet of the output quality (**(3) quality criterion**) is important. If the goal is to entertain a user, variation in the outputs, metaphoricity or humourousness might be crucial. If one needs to provide accurate description of the inputs, faithfulness to the input's content is important. A mistake would be to expect a system to produce a high-quality output, without specifying which quality dimension is important.

Researchers need a way to define an evaluation procedure that drives the system development process in the right direction. They usually rely on functions (**(4) automatic metrics**) that can be computed throughout this process, to automatically assess the quality of the system at a particular step. Unfortunately, modeling certain quality criteria (like metaphoricity or humourousness) computationally is very hard (Do Dinh and Gurevych, 2016; Simpson et al., 2019; Yu et al., 2020). In addition, metrics might have additional constraints, e.g. they might need to be differentiable, or rely on external resources, or be computed in relation to the outputs of other systems, etc.

Finally, while researchers sometimes rely on automatic metric results to compare systems, the proponents of intrinsic human-based evaluation argue that metric evaluation results should be backed up by a **(5) human evaluation** study. The main reason why it is not done as much as it is desired by the community is that human evaluation studies are costly, labour- and time-consuming. Nevertheless, automatic evaluation results are often considered unreliable, and one of the few ways to validate them is via human experiments (Belz and Reiter, 2006).

These five elements form what we call the *Sanity Polygon* (Figure 3.1). These components are necessary for any NLG experiment, and reasonable treatment of each of them contributes to the success of the study. We do not take credit for fleshing them out; there have been attempts to do it before (Dale and Mellish, 1998; Scott and Moore, 2007; Popescu-Belis, 2007). In fact, these components are not even specific to NLG — any NLP study would contain them. Our contribution lies in showing that the *interaction between the components is what makes an NLG study successful*.[8] Through literature analysis we have concluded that a failure in NLG experiments

[7] `https://en.wikipedia.org/wiki/Garbage_in,_garbage_out`

[8] Depending on the aim of the study, *success* means empirical evidence of a system's superiority compared to contenders, solid assessment of corpora quality and detection of possible data artifacts, a proof that an automatic metric is a viable proxy of human judgements of a certain quality criterion, etc.

in most cases can be attributed to some sort of a mismatch between any of the components. In the following sections we examine each of the connections in more detail and provide examples of past studies that support our claim.

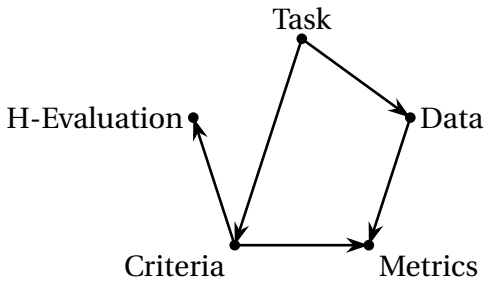


Figure 3.1: Sanity Polygon (SP): the five key components of an NLG study, and identified relations between them.

3.3 Task and Data

3.3.1 Data Annotation

Training data is the source of the statistical patterns that NLP systems learn, which is why it is so important to have diverse, clean and consistent training data. The most important factors influencing its quality include (but not limited to) the annotation source, the complexity of annotation guidelines, the annotators' motivation and the number of annotators per data point (Jurgens, 2013; Brysbaert, 2019; Van der Lee et al., 2019).

Specifics of Annotation Source *Consider the specifics of your data source and the incentives of the annotators.* Most NLG annotations assume semantic intepretation of textual data, which is very subjective. Therefore, gathering a wide range of high-quality human annotations is desirable.

Preparing *expert annotations* has been the dominating way of corpus creation for decades. Expert annotations are usually costly and labour-intensive, but are considered to be of high quality and consistency (Reiter and Sripada, 2002). By consistency we mean a high inter-annotator agreement (IAA), which is the average pairwise probability that two annotators agree, adjusted for chance (Cohen, 1960). In this sense, a higher-quality data annotation is the one in which multiple humans provide the same annotation for the same examples.

However, as data-driven models were getting larger, the data size requirements were increasing and, consequently, the cost of data annotation experiments grew significantly. Many researchers started exploring ways of delegating annotation tasks to a large group of untrained individuals, as opposed to a smaller selection of experts (Mairesse and Young, 2014). *Crowdsourcing* became a way to scale up the data creation process at the expense of a decrease in data quality. In order for the latter not to influence the overall system results, researchers started developing methods to reliably aggregate the noisy data coming from many annotators (Novikova et al., 2016).

Our literature analysis suggests that crowdsourcing is currently the dominant method of annotating NLG data. However, motivating crowdworkers is hard, because people who are motivated by monetary rewards often differ from the users who are unpaid and motivated by other means (Kuznetsov, 2006; Nov, 2007). Moreover, using money to motivate people is not easy and does not always meet expectations. Mason and Watts (2009) studied financial incentives of Amazon Mechanical Turk[9] workers and found that, surprisingly, increasing the amount of compensation for some tasks does not necessarily improve the the resulting quality. Another unexpected observation was described by Callison-Burch and Dredze (2010) who found examples of an inverse relationship between the amount of payment and the job quality of crowdworkers. Chasing after additional income lead some of them to take on tasks for which they clearly lacked qualifications. In such a situation, some crowdworkers resorted to cheating, which harmed the annotation quality. Ariely (2009) found that adding a monetary incentive can diminish the initial intrinsic motivation of the worker. Gneezy and Rustichini (2000) discovered that adding monetary compensation in a non-monetary environment leads to decrease in work performance.

Enhancing activities by creating similar experiences to those experienced when playing games in order to motivate and engage users (*gamification*) has been getting more popular in recent years (Hamari, 2019). Crowdsourced annotations from games-with-a-purpose (also called *serious games*) is an attractive alternative to paid crowdsourcing. In a serious game, the players lack the expertise to solve the underlying task using scientific methods, but do it out of free will, while playing the game. Compared to experts or crowdworkers, serious game players have a much stronger incentive to do their job well (Morschheuser et al., 2016). Successful applications of serious games in NLP include word sense labeling (Venhuizen et al., 2013),knowledge base extension (Vannella et al., 2014), answering quizzes (Ipeirotis and Gabrilovich, 2014) and fallacy detection (Habernal et al., 2017).

Ambiguity is Good *Treat ambiguity as a point of interest: instead of getting rid of the ambiguous cases in data annotations, analyze them.* Oftentimes, data contains ambiguous cases which do not assume one true answer, but allow for multiple alternatives. These cases often cause a lot of disagreement between the annotators. The latter, in turn, has been considered a sign of poor quality — either bad annotation guidelines (expert annotations), or unqualified crowdworkers without sufficient training. However, IAA depends on many factors, including their aptitude for the task, how much they are paying attention, how much guidance they are given and how much of the guidance they are able to remember (Manning, 2011).

More importantly, there is empirical evidence that disagreement is an important signal of ambiguity inherent to the data sample, an interesting phenomenon that demands closer investigation (Aroyo and Welty, 2015). These data points turn out to be challenging and most interesting from a data analysis point of view. The right course of action would be not to throw the data points out, but separate them into a group of challenging cases and report about them.

[9] A web platform commonly used by NLP researchers to crowdsource annotations: `https://www.mturk.com`.

Complex Guidelines are Bad *Do not make the guidelines too complex or detailed.* One of the ways to improve IAA is via adding more detailed annotation guidelines. This is generally considered as good practice, since it addresses many of the corner cases that cause annotators to disagree, and thus potentially decrease the IAA.

However, is has also been found that too precise annotation guidelines increase the agreement by forcing expert annotators to make choices they may not actually think are valid, and removing the potential signal on individual examples that are vague or ambiguous (Aroyo and Welty, 2015). Crowdsourcing tasks also do not allow long, complex annotation guidelines, which forces one to keep instructions simple and precise. This has several benefits: it reduces the time spent on annotation task design, allows annotators to make choices they are more comfortable with, and reduces annotator training time.

Annotation Redundancy *Assign multiple annotators for each data point and create redundant annotations.* NLP data annotation does not always assume just one perspective. Language is ambiguous by nature, therefore in many cases there are multiple valid interpretations of a data point. Multiple assessors' annotations might be considered redundant, but they allow the task designer to catch ambiguous cases and estimate the difficulty of various data points. That is why multiple workers should be presented the same object of interpretation.

Choice of Annotators *Experts are not always better than non-experts.* There are studies that showed that expert annotations are not always qualitatively better than those coming from layman annotators. For example, Aroyo and Welty (2015) found that mistakes by the crowd are often not surprising, while experts are far more likely to see a target relation between entities, where none are expressed in a sentence, when they knew the relation to be true. Studies of tag-searching behavior of crowdworkers showed that only a small fraction of the tags could be found in the documentation created by professionals, which means that experts and professionals often search for different things (Leason, 2009; Hildebrand et al., 2013). This, perhaps, can be explained by an informal argument that experts often operate in a "sandbox" scientific environment, while crowdworkers represent the real-world audience of the developed technique.

Mind Data Life-cycle *Consider the presence of time-dependent phenomena in your data.* If you are using a well-established dataset for your experiments, mind two important caveats:

- over time, the community overfits to the dataset, which overestimates the generalization abilities of the proposed approaches;
- over time, some annotated phenomena change and annotations become outdated.

To illustrate the first point, we refer to the work of Manning (2011), who was discussing the necessity and possibility of improving the accuracy of modern POS taggers beyond 97 %.[10] The author admits that it might be important for researchers to have static versions of data for the sake of comparable experiments. However, he also warns against the negative tendency of researchers

[10]Token-level accuracy of Stanford POS tagger on Penn Treebank data.

to keep the datasets "frozen" in time. The author provides examples showing that many tagging errors stem from the inconsistencies in the data annotation, and the way to improve further is to either improve the taxonomy of tags to allow clearer automatic tagging, or improve the conventions by which the tags are assigned.

A real danger of working with the data that has been around for too long is a natural tendency of researchers to overfit to this data. Since beating the state-of-the-art results admits one to publishing a paper in prestigious conference proceedings, many researchers chase after this ideal. This leads to a situation when even the test set is no longer "blind": since many perform error analysis of system predictions on the test set, by the time the 20th researcher publishes their score on the leaderboard, everyone is already aware of the test data challenges. Next proposed approaches are then built to overcome these challenges and push the scores higher, which can be considered as a type of overfitting to test data distribution.

The second mentioned issue is pertaining the semantic changes of various phenomena targeted in NLP systems. A great example was given by Aroyo and Welty (2015):

> *[Osama bin Laden] used money from his own construction company to suport the [Muhajadeen] in Afganistan against Soviet forces.*

In a task of identifying mentions of terrorists in a document, this example will have a different treatment, depending on when the annotation takes place. In the 1990s, for example, the *[Osama bin Laden]* part would have been labeled as *HERO*, but after 2001 — as *TERRORIST*. Both entity types would be valid for the same phrase, even though they introduce conflicting roles for the same entity. This means, that both corpora need to be adapted over time to reflect such semantic changes. Researchers need to be aware of such phenomena and develop approaches with this issue in mind, because this issue is one of the causes of data bias. The usage of lexis which once was considered normal, at some point in time may be considered as offensive. If an outdated corpus is used, these undesirable artifacts may propagate into a NLG model.

An example of temporal data shift in spoken language understanding is described by Kim et al. (2017). The authors show that a model trained on data from 2013 cannot properly handle data from 2014–2016, because the content of the user queries changes over time (e.g. new restaurant or movie names may be added). Consequently, the model performance degrades over time.

3.3.2 Data Artifacts

The statistical patterns that data-driven models learn from data include *noise* and undesirable *biases*. Informally speaking, noise is *deviations from what is considered to be representative for a phenomenon*. In the case of Natural Language Processing, noise examples include spelling errors, irregularities and idiosyncrasies in the use of punctuation, white space and capitalization, content differences between source and target documents, non-literal translations, errors in document alignments, etc (Clark and Tim, 2003; Khayrallah and Koehn, 2018; Khadivi and Ney, 2005).

Bias is an umbrella term denoting differences between an intended data distribution and the distribution used or produced by the model (Shah et al., 2020).

Noisy Data Treatment *Noisy data should receive special treatment — do not leave the data discrepancies unattended.* An important facet of the task-data relationship is treatment of noise contamination. It has been noticed long ago that data noise always finds its way into the generated text (Scott and Moore, 2007). It is important to bear in mind though, that *noise does not necessarily mean errors propagating into the training set with imperfect annotations*. It can also be part of the true data distribution: for example, in cases where the domain assumes irregular language usage, like colloquial speech on the web. And if the task is to generate colloquial speech, then sanitizing the outputs might have an adverse effect.

If one's task is to generate refined language from noisy data, they need to find ways of reducing the influence of noise on the final model predictions. One way is to retract back to templates and rules. But in case the complexity of a task does not permit using rules, one has the option of employing an approach which can detect noisy instances (or noise sources) and recover true labels (Hovy et al., 2013).

Bias Considerations *Consider the possible presence of bias in your data, especially if you use it for production-level models.*

An important point of consideration is the possible existence of bias in your data. The sample bias phenomenon, most relevant from a NLP perspective, has been in the focus of social scientists for a long time (Berk, 1983). Until recently, NLP researchers have been predominantly occupied with biased predictive models in the context of domain adaptation (Jiang and Zhai, 2007). The community tried to understand why models fit on data from one domain do not perform well on other text types, and proposed various methods to mitigate the issue.

However, democratization of machine learning made it possible to use NLP models in many areas of everyday life. This inevitably caused many unexpected issues. The infamous Tay bot, an AI chat robot deployed by Microsoft to Twitter, transformed into an "evil Hitler-loving, incestual sex-promoting, 'Bush did 9/11'-proclaiming robot" in just one day of interaction with the users.[11] The developers created the bot in order to improve the customer service on their voice recognition software, but they did not expect the users to abuse the bot and teach it bad things. The bias that users were injecting into the data while interacting with the bot, caused the model to generate rather improper tweets, and Microsoft had to shut it down.

There are many more examples how data bias becomes the source of algorithmic discrimination: in criminal risk assessment,[12] predictive policing,[13] credit eligibility estimation,[14] human resource management,[15] etc.

Data bias exacerbates inequality and causes serious issues with long-lasting consequences. Those who design machine learning algorithms and train them share the responsibility of those

[11] https://www.telegraph.co.uk/technology/2016/03/24/microsofts-teen-girl-ai-turns-into-a-hitler-loving-sex-robot-wit/

[12] https://www.propublica.org/article/machine-bias-risk-assessments-in-criminal-sentencing

[13] https://rss.onlinelibrary.wiley.com/doi/full/10.1111/j.1740-9713.2016.00960.x

[14] https://www.brookings.edu/research/credit-denial-in-the-age-of-ai/

[15] https://qz.com/work/1098954/ai-is-the-future-of-hiring-but-it-could-introduce-bias-if-were-not-careful/

who deploy the models. Researchers have identified different bias types, and proposed various methods to tackle them (Li et al., 2018; Elazar and Goldberg, 2018; Coavoux et al., 2018). Shah et al. (2020) recently proposed a conceptual framework outlining and relating the different origins of bias. According to the authors, the variety of biases indentified so far can be categorized as follows:

- Label bias: when the distribution of the dependent variable in the data source diverges substantially from the ideal distribution.
- Selection bias: when some members of the intended population have a lower or higher sampling probability than others.
- Overamplification: when a trained model relies on a small difference between attributes with respect to the objective, but amplifies this difference to be much more pronounced in the predicted outcomes.
- Semantic bias: unintended associations and societal stereotypes picked up as a side-effect of distributional semantics.

Note that these sources are distinct, but arise due to the same reason — the differences between an intended distribution and the distribution produced by the trained model. These sources of bias pose a real threat to any data-driven endeavor. The best way to deal with them is to do data analysis, attempt to identify biases in your data and develop respective countermeasures.

As a form of summary, we construct a requirements checklist which is meant to be consulted when analyzing the task at hand. The checklist below shows important considerations concerning the relation between the task specification and the data collected for training, as well as evaluating NLG approaches.

Task and Data

- Data annotation:
  - ☐ Is the phenomenon at hand ambiguous?
  - ☐ Are the annotation guidelines simple?
  - ☐ Are the guidelines ambiguous?
  - ☐ Do annotators have the needed expertise?
  - ☐ How many annotators are there per data point?
  - ☐ Does the data contain time-dependent phenomena?
- Data artifacts:
  - ☐ Is the data noisy?
  - ☐ Could the data be biased?
  - ☐ Is it a low-resource scenario?

3.4 Task and Evaluation Criteria

3.4.1 Task-Criteria Match

Task-specific Criteria *Evaluation criteria should be task-specific, targeting quality facets important for the task.* Evaluation criteria are defined as part of the experiment design and correspond to a specific task and type of evaluation (Section 2.4.1). For example, it makes little sense to measure semantic correctness in the task of language modeling, since there is no one and only target output that should be generated by a trained model.

A more common mistake is to not define criteria which should be defined prior to conducting experiments. This issue often bubbles up in human evaluations, when the assessors are not being asked about a quality facet that was defined as part of task specifications. For example, consider a task of image caption generation. The variability of captions is often mentioned as a challenging property of the task, but is rarely measured in the human experiments (Xu et al., 2015a; Kilickaya et al., 2017).

3.4.2 Criteria Scope

Atomic criteria *Evaluation criteria should be focusing on just one quality facet.* Intuitively, one wants their system to produce good outputs. Unfortunately, as of today, NLG community does not have agreed objective criteria for comparing the *goodness* of texts (Dale and Mellish, 1998; Scott and Moore, 2007). Consider possible definitions of *accuracy* and *fluency* from (Dale and Mellish, 1998):

- Accuracy: does the text convey the information it is supposed to?
- Fluency: does the text present the information in the readable manner?

The authors mention that these are commonly used informal measures of text quality. But they also admit that there is a "general uncertainty about what the key dimensions of measurement" which these criteria capture. We conjecture that the debate around objectivity and subjectivity of the evaluation criteria is connected to their scope, or assessment granularity. Subjective terms like *appropriateness* or *correctness* that are sometimes used to give a qualitative assessment of a system output, cannot be used to objectively evaluate NLG outputs, since these criteria depend on the task performed and the definition of what constitutes success. Other examples of subjective qualities include *good style*, *coherence*, *readability* (Dale and Mellish, 1998).

A comparative study of the evaluation methodology for NLG in interactive systems, conducted by Hastie and Belz (2014), also concludes: when designing evaluation experiments "it does not make sense to pair up individual evaluation measures with single abstract, higher-level quality criteria". The latter include (among others):

- dialogue performance;
- dialogue quality;
- system usability;

- user satisfaction;
- efficiency.

Reducing the scope of a criterion is not an easy task. Sometimes this leads to a total failure of the experiment. For example, Dale and Mellish (1998) cites the work of Carter (1996) on the automatic generation of hypertext. The authors of the latter work chose to measure mean time per node as an indication of how easy it was to navigate through a hypertext system. In other words, the criterion was *navigation ease* and *mean time* was the metric: low mean time indicated easy navigability. It turned out that certain subjects, when lost, resorted to rapid random clicking and this unexpected phenomenon rendered the measure, as well as the collected user data, useless.

The requirements checklist for the task-criteria relation looks as follows:

Task and Evaluation Criteria

- Task-Criteria Match:
  - ☐ Do the criteria reflect the specifics of the task?
- Criteria scope:
  - ☐ Is each individual criterion focused on a single quality dimension?

3.5 Evaluation Criteria and Human Experiments

Mismatches between evaluation criteria and human evaluation often manifest themselves as inconclusive evaluation results or, in the case of a correlation study, low correlation coefficient values between automatic metrics and human assessments. In most cases the problem is caused by the failure of human evaluators to understand precisely what they need to assess.

3.5.1 Human Experiment Instructions

Explicit Definitions *Evaluation criteria should have definitions explaining what the criteria measure.* Sometimes researchers use an evaluation criterion in their experiments, but leave its definition out, perhaps assuming that for a human it is clear which quality dimension is being captured. For example, human evaluation experiments conducted by Reiter and Belz (2009) included *clarity* and *readability* as an assessment of linguistic quality and *accuracy* and *appropriateness* as an assessment of content quality. The authors, however, did not define any of these terms precisely (at least the respective paper does not say anything about that). When analyzing the results, they observed a significant correlation between the *accuracy* and *clarity* scores that the test subjects

gave to texts. As the authors note, this could have been caused by the subjects not being able to distinguish *accuracy* from *clarity*. We conjecture that this was largely due to the absence of the criteria definition.

Precise Definitions *Evaluation criteria should be defined as precisely as possible.* Sometimes it is not clear what a criterion means, precisely. Consider the definition of *fluency* which is commonly used as one of the evaluation criteria in NLG research. We found it to be probably one of the most overloaded terms in NLG literature. Merriam-Webster dictionary defines someone being fluent as *capable of using a language easily and accurately*.[16] Cambridge dictionary describes fluency as *the ability to speak or write a language easily, well, and quickly*.[17] The term is generally well understood by native speakers of English, which we confirmed by an informal survey of native speakers in our own research lab. However, none of the researchers we asked was able to give a precise description of what it means to be fluent. Non-native speakers have more trouble defining the term. Our understanding is that fluency is a composite characteristic of one's language usage, but not a single property which can be used to assess a text snippet across a specific quality dimension.

The absence of a precise definition of an evaluation criterion leads to a divergence of its interpretation. According to a recent study of 37 research papers published as part of the proceedings of the Association for Computational Linguistics on evaluation methodologies in automatic question generation, more than twenty different quality criteria have been used by researchers in the period from 2013 to 2018 (Amidei et al., 2018):

- grammaticality
- semantic correctness
- answer existence
- naturalness
- question type
- clarity
- discriminator quality
- relevance
- correctness
- well-formedness
- key selection accuracy
- corrected retrieval
- fluency
- coherence
- timing
- inference step
- question diversity
- importance

[16] `https://www.merriam-webster.com/dictionary/fluency`
[17] `https://dictionary.cambridge.org/dictionary/english/fluency`

- specificity
- predicate identification
- difficulty
- overall criterion

In some studies evaluation criteria are being merged, which further complicates the analysis of the results. For example, Nenkova et al. (2010) conducted a study on the predictive power of structural features in measuring the linguistic quality of texts in machine translation and document summarization. The study uses the terms *readability* and *fluency* interchangeably, which makes it hard to understand which criterion ultimately the metrics correlate with.

This situation is thought-provoking, because evaluation of automatic question generators involves assessing just two major quality components: *grammatical correctness* and *semantic correctness*, i.e. the question should adhere to the grammar of the target language and be about the content described in the answer. Yet, the large number of criteria mentioned above means that either researchers understood the task differently, or they were being careless in defining the criteria, which ultimately lead to a confusion.

Whenever a human assessor encounters a generic evaluation criterion, we face the problem that humans will not agree about subjective qualities (Dale and Mellish, 1998). One might hope to avoid this problem by having sufficient independent judgements: in this case, disagreements will become invisible as a result of the averaging process. But then a sufficiently large number of judgements should be collected. However, what constitutes "sufficient" depends on the complexity of the task at hand.

In our opinion, a viable solution to this issue could be to *explicitly provide criteria definitions in the study*. Perhaps, ultimately it does not even matter what the criterion is called — one could call it *Criterion E*, for example — as long as the definition is clear, documented and publicly accessible.

Criteria Independence *Evaluation criteria should not overlap.* Correlating or overlapping definitions of evaluation criteria often result in an inability of human assessors to distinguish between the criteria used in a human experiment (Novikova et al., 2017b). Such dependence ultimately renders human evaluation results useless.

Right Elicitation Method *Choose the elicitation method during the experimental design (not at the time of analysis). If in doubt, use preference-based methods.*

When planning human evaluation, one has several methods of human judgment elicitation at his or her disposal (Section 2.4.3). However, there is little guidance in terms of various trade-offs different methods have.

For example, due to the simplicity of the design, discrete scales became the method of choice in many NLG evaluations. However, Amidei et al. (2019) reviewed the use of rating and Likert scales for NLG evaluation over the last ten years and discovered that many NLG studies confuse Likert scales and rating scales. Often Likert scales are used for an item-by-item analysis, which is wrong: aggregate scales, such as Likert scales, are created to estimate the overall opinion of a responder about some phenomenon by use of aggregate items. A common mistake is to extract

items from an aggregate scale, in order to present a more fine-grained evaluation. Amidei et al. (2019) warn that such evaluation can lose its meaning and lead to false conclusions.

Another suggestion from Amidei et al. (2019) is to be cautious when using parametric statistics to analyze the results obtained using scales. There are many things that could go wrong when applying parametric statistics on data which are not interval (Liddell and Kruschke, 2018). The general advice is to do preliminary verification of the parametric statistic assumptions, prior to conducting such analysis.

It is important to consider task simplicity from the perspective of the human assessor. In this regard, we favor preference-based evaluation, since in many cases it is much easier to express a preference between two items, rather than score them both on a scale.

3.5.2 Human Experiment Difficulty

Human assessors are influenced by too many factors which sometimes cannot be controlled. When analyzing the shortcomings of BLEU, NIST and METEOR metrics, Turian et al. (2003) pointed out something obvious, but very important:

> Automatic MT evaluation measures cannot be faulted for poor correlation with the human judges, as the judges do not correlate well with each other.

In order to make the best of human evaluations we proposed the following recommendations.

Simplify Evaluation Tasks *Design simple tasks, since they lower the cognitive load for the assessors.* It is generally recommended to consider the cognitive difficulty of the task and ways to make it easier for the human assessors. This is usually done in order to make the experiment results to degrade gracefully. Dale and Mellish (1998) analyze an experiment of Yeh and Mellish (1997) who compared algorithms for the generation of anaphoric referring expressions in Chinese by measuring the extent to which the generated references agreed with those selected by humans. Dale and Mellish (1998) note that the humans could not always agree, and speculate whether ranking their selections instead of indicating the best would be a measure of agreement, more robust with respect to certain random variations. Whether this is true or not, is not clear, since the experiment has not been reproduced with the new proposed criterion, but this is a good example of how one might attempt to achieve better correlation results by simply reducing the difficulty of the annotation task.

The requirements checklist for the relation between human experiments and evaluation criteria is given below.

- Human Experiment Instructions:
  - ☐ Are evaluation criteria defined in the experiment?

- ☐ Are the criteria definitions precise?
- ☐ Are the defined criteria independent?
- ☐ Which human judgment elicitation method is used?

- Human Experiment Difficulty:
  - ☐ Is the evaluation task simple?

3.6 Evaluation Criteria and Automatic Metrics

The studies that we conducted ourselves, as well as the ones we examined as part of the literature analysis, make us hypothesize that in many cases the low correlation between evaluation criteria and metric scores is caused by a semantic mismatch between them. By semantic mismatch we mean a *disparity in what the metric was designed to do, and what the criterion measures*.

3.6.1 Targeted Language Level

Correct Linguistic Level *Choose metrics according to the evaluated linguistic level.* Depending on textual and real-world context, system outputs can differ from the source text at any linguistic level (lexical, syntactic, semantic, discourse) and still be considered perfectly correct (Fomicheva and Specia, 2016).

A study conducted by Novikova et al. (2017b), revealed that some important subtasks of NLG, such as sentence planning and text structuring, are almost not taken into account by BLEU and similar metrics. This seems to be expected, since these metrics were not designed to measure text coherency. Nevertheless, researchers still continue to use them for the overall system evaluation. This mainly happens when an NLG component is evaluated as part of a bigger system (e.g., text summarizer), in which case content selection is viewed as more important than grammaticality or overall text structuring.

Barzilay and McKeown (2005) observed that humans are very sensitive to processing ungrammatical sentences, and when being asked about the quality of the text, often consider whether it is grammatical or not. N-gram overlap metrics are bound to fail as proxies for such a criterion. Yet, some researchers tried using them to evaluate grammaticality of image captions, but, as expected, failed to find any significant correlation with human judgements of grammaticality (Elliott and Keller, 2014).

3.6.2 Task Specificity and Metrics

Task-Metric Match *Choose metrics that reflect the task specificity and the defined criteria.* In other words, one needs to find the metrics that make sense as proxies of the respective criteria.

NLG as a field has only started to develop its own metrics. Until recently, NLG researchers in their experiments simply reused metrics designed for other tasks. For example, BLEU, NIST and METEOR came from the machine translation domain, ROUGE variants — from document summarization. However, numerous human evaluation studies showed that system comparison based on metric scores is not reliable for many NLG processes (Scott and Moore, 2007; Reiter and Belz, 2009). We provide one plausible explanation for this below.

In contrast to machine translation, syntactic parsing or POS tagging, NLG is not a monolithic task, which is why the practice of comparing outputs of systems which were designed to do different things seems questionable. A metric designed for one task does not necessarily work as a drop-in replacement in another.

Consider BLEU as an example metric. The original paper that introduced it showed that BLEU correlated well with human judgments of translation quality. The study also showed the ability of the metric to distinguish between human and machine translations (Papineni et al., 2002). This correlation has been further confirmed in the annual NIST Machine Translation Evaluation exercise. The usage of BLEU in machine translation was later criticized by Callison-Burch et al. (2006) who have found theoretical and practical evidence that BLEU might be a wrong metric to be used in several setups, including those where one is trying to detect improvements for aspects of translation that are not modeled well by BLEU (e.g. *grammaticality*), and monitoring improvements that occur infrequently within a test corpus. As described in Section 2.4.2.1, BLEU is a corpus-level metric which assesses the accuracy of translations in a multi-reference setting. Using multiple references is supposed to provide support for variability of the translation process. However, many NLG tasks assume a much larger valid candidate output pool. For example, in the task of image caption generation or style transfer, one cannot hope to cover a significant portion of the valid candidate search space for a given input with the number of references commonly used in machine translation, which is around four (Belz and Reiter, 2006; Qin and Specia, 2015).

A study by Giménez and Màrquez (2007) showed that "n-gram overlap is neither necessary nor sufficient for two sentences to convey the same meaning". Consider the following example from a recent study on evaluating image caption generation approaches using automatic metrics (Anderson et al., 2016):

A young girl standing on top of a tennis court.
A giraffe standing on top of a green field.

These two captions describe two different images from the MS COCO dataset (Lin et al., 2014). However, comparing the captions using any of the wildly used n-gram overlap metrics would give a high similarity score due to the presence of the long 5-gram phrase 'standing on top of a' in both captions.

The second example from this study compares two captions obtained from the same image:

A shiny metal pot filled with some diced veggies.
The pan on the stove has chopped vegetables in it.

Despite the fact that these captions convey almost the same meaning, they exhibit low n-gram similarity as they have no words in common. Note that the reason why n-gram metrics are a bad

choice for evaluating image caption generation systems is not because the former are inherently flawed. The reason is that the task expects a high variation in the output space, which cannot be captured by metrics computing n-gram overlap with references.

The requirements checklist for the relation between the evaluation criteria and automatic metrics is as follows:

Evaluation Criteria and Metrics

- Targeted Language Level:
  - ☐ Does the metric measure the correct linguistic level?
- Task Specificity and Metrics:
  - ☐ Is the metric supposed to approximate the chosen evaluation criterion?

3.7 Data and Automatic Metrics

The goal of this section is to draw attention to metric assumptions, or conditions, that should hold for it to work as expected.

3.7.1 Metric Assumptions

System Output Quality *Some metrics have strong assumptions about the system output quality. Metric evaluation is unreliable when these assumptions are not met.* Many metrics have other assumptions which are often left unattended. For example, the ORANGE metric used to evaluate machine translation metrics, proposed by Lin and Och (2004), has the assumption that automatic translations are worse than their reference translations. The metric incorporates this assumption into score computation. Given a source sentence, an n-best list of its machine translations, and its reference translations, the average rank of the reference translations within the combined machine and reference translation list is computed. One then ranks these translations, calculates the average rank of the references in the n-best list, and computes the ratio of the average reference rank to the length of the n-best list; *the smaller the ratio is, the better the automatic metric is*. Clearly, as the quality of the outputs improves, the assumption stops working. And while ORANGE is used to evaluate metrics (not system outputs), the conclusion is still valid, since similar protocols are used by automatic metrics when evaluating the quality of model predictions.

Many studies ignore the fact that, for example, the original paper which introduced BLEU showed that it can only distinguish between very good and bad translations (Papineni et al., 2002). Belz and Reiter (2006) also suggests that BLEU is not effective at evaluating texts which

are as good as (or better than) the reference texts. The authors also note that this might not have been a problem earlier, because in the early 2000s outputs of wide-coverage MT systems were generally worse than human translations. But NLG systems (even then) were often domain-specific and were able to generate texts that were judged better by humans than human-written texts.

The results of the experiments conducted by Babych and Hartley (2008) on machine translation system outputs serve as an additional evidence for this point. The authors showed that proximity-based metrics (in particular, BLEU) loose sensitivity as the systems become more accurate, but performance-based metrics (e.g. accuracy of a named entity recognition system facilitated by machine translation outputs) remain sensitive across the scale. The authors further attribute the stable sensitivity of performance-based metrics to measuring the cumulative functional effect of different language levels, while proximity-based metrics measure structural matches at a lexical level only and therefore miss higher-level errors that are more typical for better machine translation systems. The authors of the study also note that task-based performance better captures legitimate variation (as opposed to spurious variation like the one we mentioned above when talking about BLEU). This is because they do not make explicit assumptions about particular combinations of structural features that perform external textual functions. Similar findings have been observed by Liu et al. (2016) and Ma et al. (2019a).

Further, (Sun, 2010) found that only when the metric score difference between the systems is greater than a certain threshold value, will the majority of human evaluators agree with the judgement of the automatic metrics. They also found that when two automatic scores of two translations are the same, it does not always mean there is no qualitative difference between the translations. Finally, perhaps the most important issue with metric-based evaluation is that it does not say anything about the real-world effectiveness of the system under evaluation. Rare exceptions (Reiter et al., 2003; Belz and Gatt, 2008) make it clear that a solid assessment of an NLG approach with respect to a certain task or downstream application is very important.

A recent study of Mathur et al. (2020) showed that outlier systems, i.e. those systems whose quality is much higher or lower than the rest of the systems, can have a disproportionate effect on the computed correlation of metrics. The resulting high values of correlation can then lead to to false confidence in the reliability of metrics.

Correct Assessment Level *Some metrics are designed to work on corpus-level; using them on segment level does not guarantee adequate evaluation results.* Another assumption that is sometimes ignored is that of the assessment level. For example, in machine translation, comparisons of automatic evaluation metrics are often conducted on a *corpus level* using correlation statistics (Pearson's product moment correlation coefficient or Spearman's rank order correlation coefficient) between human scores and automatic scores. A possible reason for this is that sentence-level lexical overlap metrics suffer from feature sparsity issues: for example, counts of higher order n-grams are usually rather small (Stanojević and Sima'an, 2014). While several prominent segment-level metrics appeared (like BLEURT and BERTscore), NLG researchers often continue to use corpus-level machine translation metrics for segment-level assessment, ignoring the fact that metrics like BLEU or NIST were designed to work on a corpus level (Fomicheva and Specia, 2019). In such

cases, metric results cannot be fully trusted, since the metrics operate under a non-native regime in which their performance becomes unstable (Chaganty et al., 2018).

Cross-task Metrics *Be cautious about the metrics that migrated from other fields and be critical about correlation study analyses in other areas, even if they were done using similar quality criteria.* Metrics that migrated from one research field to another, even when being evaluated on similar criteria, might not work, because the performance of automatic metrics in terms of human vs. system correlation analysis is not stable across different evaluation settings (Lin and Och, 2004). In other words, if it has been found that BLEU-4[18] correlates well with human judgements of fluency in machine translation, the same might not be the case for NLG. Often researchers simply claim that BLEU does or does not correlate with human judgements — that is very inaccurate on many accounts and should be avoided.

3.7.2 References

High Reference Quality *The majority of metrics assume access to high-quality references.* Automatic metrics are a proxy method for evaluating the quality of system outputs. This is why one basic assumption of all automatic evaluation metrics is that references are of high quality and the more an output is similar to its references the better. That is not necessarily the case: with the proliferation of crowdsourcing, human evaluations are losing their reliability, as crowd-workers are hard to manage, which leads to various quality control problems (Daniel et al., 2018). This causes a situation when the gold standard data contains a lot of noise, which necessitates an additional data curation step before the annotations can be used for system evaluation (Freitag et al., 2020).

Sufficient Number of References *Make sure that the number of references is sufficient for reliable metric evaluation.* The number of references needed to make evaluation meaningful depends on several factors. Studies have shown that automatic metrics by design are not equally sensitive to the number of target references. For example, TER has been shown to give competitive results to BLEU, while using only 25 % of the available references (Snover et al., 2006). Finch et al. (2004) showed that machine translation evaluation performance usually improves with increasing numbers of references, although with diminishing returns. For example, adding up to sixteen references improves the correlation for BLEU. However, little improvement has been seen from adding more than four references when scores were computed by GMT; NIST only showed improvements with small numbers of references (1–4), and adding more references degraded its performance (Figure 3.2).

It is not necessarily the case that the more references one has, the better. For example, Doddington (2002) found that increasing the number of references yields only modest improvements in evaluation performance in the machine translation task on the DARPA corpora.

[18]The version of BLEU that counts n-grams up to the fourth order.

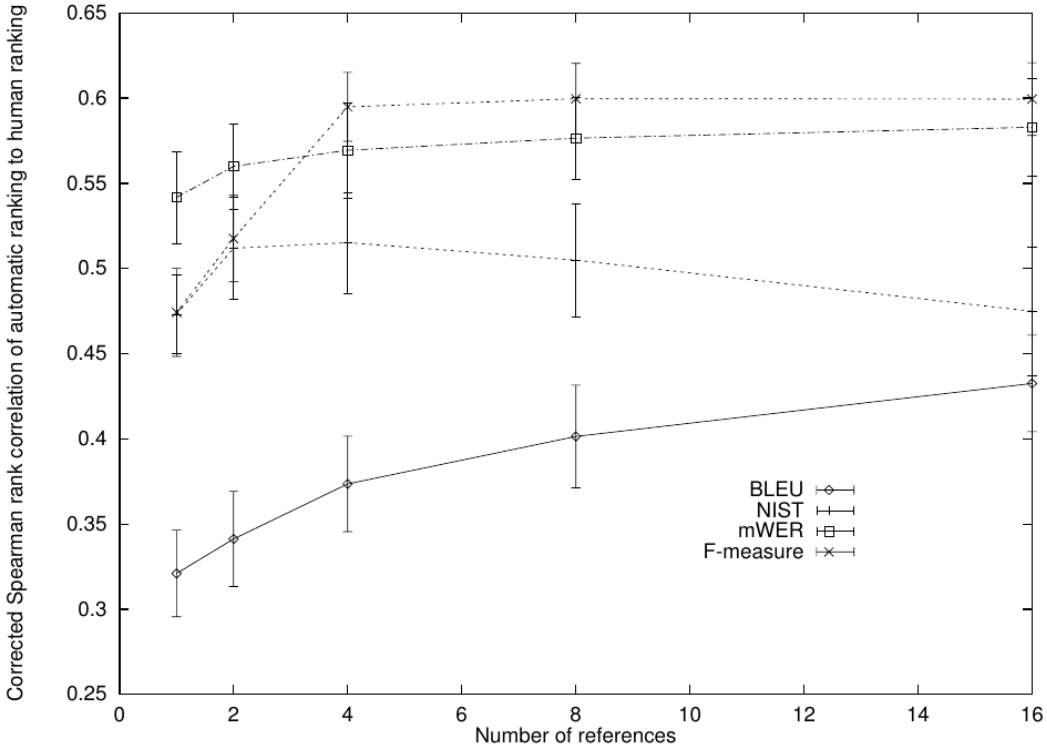


Figure 3.2: The effect of adding more references on automatic metric evaluation in machine translation. Figure taken from (Finch et al., 2004).

Metric comparisons rely on human judgments of the output quality, but the judgments themselves are often inconsistent and very expensive to acquire (Lin and Och, 2004). This means that a larger number of references is needed, when noise is inherent to the dataset.

The number of references in a good corpus also depends on the expected language variability in the output texts. In tasks like caption generation, document summarization, style transfer, a large number of variable outputs is desirable, but this also depends on the length of the references (since longer texts naturally exhibit more variability).

The requirements checklist for the relation between data and automatic metrics is given below.

Data and Metrics

- Metric Assumptions:
  - ☐ Does the metric assume low quality of the outputs?
  - ☐ Is the metric supposed to work on corpus/segment level?
  - ☐ Has the metric migrated from another task?

- References:
  - ☐ Is the metric sensitive to the reference quality?
  - ☐ Is the number of references sufficient?

3.8 Other Metric Considerations

This section contains various considerations pertaining the choice of automatic metrics in the experiments. These considerations are not about a relation between the components of the Sanity Polygon. They are about metric properties and are advised to be taken into consideration when planning the experiments.

Approach Dependence *Some metrics do not work as expected, when comparing typologically different systems.* Here, by *system type* we mean the employed inference engine (Section 2.3.1), i.e. if the system is a variant of a rule-based, data-driven or hybrid type. The reliability of automatic word-based metrics is becoming more and more questionable. Recent studies found that very often such metrics measure data- and system-specific properties of different approaches (Novikova et al., 2017b). In a study of the role of BLEU in machine translation research, Callison-Burch et al. (2006) discovered an interesting discrepancy between BLEU scores of machine translation system outputs and human judgements of their adequacy. The discrepancy manifested itself as an outlier reaching a low BLEU score, but high human assessment judgement. The system was not fully automatic machine translation, it was a post-edited manual selection of translated phrases. Subsequent experiments were conducted to verify the finding. They showed that BLEU may not be appropriate for comparing systems which employ different translation strategies. This, in turn, opens the possibility that in order for BLEU and similar metrics to be valid, only sufficiently similar systems should be compared with one another.

A recent study of Fomicheva and Specia (2019) compared how well common machine translation metrics correlate with human assessments when evaluating neural vs. statistical MT systems. They found that metric evaluation achieves a higher correlation with human assessments when evaluating the outputs of neural MT systems. Subsequent error analysis showed that neural MT systems make more adequacy errors which are more easily detected by evaluation metrics, compared to the ones affecting translation fluency.

Fragile Correlation Studies *Be critical about correlation studies.* It is important to understand the connection between the instructions for human assessors, evaluation criteria and chosen automatic metrics to draw valid conclusions about metric scores. Our literature survey showed that NLG researchers found evidence of both the existence and absence of correlation between automatic metrics and human judgements of some particular textual quality dimension. Obviously,

this does not help when one needs to determine which metrics to use for the task, or how to design a human experiment to determine the correlation.

We hypothesize that there are two main causes for the falsely low correlation between human judgements and metric scores. The first one has already been described above — it is a situation when the metrics do not match the evaluation criteria. For example, BLEU is bound to perform poorly when measuring humourousness of system outputs. The second situation is when the metric is chosen according to the evaluation criterion definition, but the definition in the human experiment instructions does not use the same evaluation criterion when the metric was chosen. A hypothetical example: the evaluation criterion is *informativeness*, ROUGE is the chosen metric, but in the human experiment the assessors are asked to rate the *quality* of the summaries. Clearly, this criterion includes many different facets. It is possible that some assessors will focus more on grammaticality, while others will consider the summaries' informativeness. In such a setup one cannot hope to get a high correlation between human assessments and metric scores.

For example, Elliott and Keller (2014) conducted a study of the correlation of automatic metrics with human judgements for the task of image caption generation. They computed correlation between automatic metric scores and human judgements of semantic and grammatical correctness of the captions. In terms of semantic correctness, the correlation varied from weak to moderately high, depending on the metric. The authors found no significant correlation between grammatlicality judgements and any of the automatic measures.

The authors further contrast these results with the study of Reiter and Belz (2009) who found no significant correlations of automatic metrics with human judgements of the accuracy of machine-generated weather forecasts. However, the latter work did find significant correlations of automatic metrics with fluency judgements.

With these two studies at hand, one might be bewildered by the conflicting results. Note, however, that the studies were different in several ways. The first one used sentence-level Spearman's ρ coefficient to measure the correlation, while the latter used document-level Pearson's r. The former chose grammatical correctness as a criterion, while the latter was evaluating fluency. The former's task was image caption generation, while the latter designed a system to generate weather descriptions from data tables. In the former case the data was crowdsourced, while in the latter it was created by human experts. Given all these differences, we conclude that the two studies were too different to put them on the same line.

To make sense of the correlation studies, one has to critically assess them and extract reliable information from them. For example, we tend to believe the study of Elliott and Keller (2014) in terms of the authors not being able to find correlation between the human judgements of fluency and automatic metric scores, because string-matching metrics are a bad proxy for that. A perplexity score computed by an in-domain language model would be a much better option. On the other hand, choosing precision-based metrics like BLEU or TER does not make much sense for a task like image caption generation, where a recall-based metric is more suitable, given the high expected variation in the outputs. In this regard, it is not surprising that METEOR and ROUGE achieved the highest correlation with human judgments of semantic correctness. Consider another fact: METEOR is a metric that was conceived to improve correlation with human judgments of machine translation quality at the segment level (Banerjee and Lavie, 2005). In light of these

facts, it is not surprising that METEOR achieved the highest correlation in the study of Elliott and Keller (2014).

Below is a short checklist which contains important considerations concerning the choice of automatic metrics in an NLG study.

Metric Considerations

- ☐ Are the compared approaches typologically different?
- ☐ Is there any evidence that the metric correlates well with human judgements of the measured quality in the task at hand?

3.9 Framework Checklist

To sum up, the aforementioned principles form a requirements checklist which we proposed to consult when analyzing the task at hand. In the subsequent chapters we are going to use this checklist to address the research questions outlined in Chapter 1.

Usage Notes Usually checklists are used in planning activities to mark the current progress. Although the purpose of SPF is different, we found this form to be rather intuitive. The idea is to go through the list and answer the questions, based on the available information. If the respective answer is seen as problematic, i.e. violating the relation between the study components as described in this chapter, we color it in red, like so: ☑. If the answer suggests that the relation under consideration is not violated, we mark it like so: ☑.

Task and Data

- Data annotation:
  - ☐ Is the phenomenon at hand ambiguous?
  - ☐ Are the annotation guidelines simple?
  - ☐ Are the guidelines ambiguous?
  - ☐ Do annotators have the needed expertise?
  - ☐ How many annotators are there per data point?
  - ☐ Does the data contain time-dependent phenomena?

- Data artifacts:
  - ☐ Is the data noisy?
  - ☐ Could the data be biased?
  - ☐ Is it a low-resource scenario?

Task and Evaluation Criteria

- Task-Criteria Match:
  - ☐ Do the criteria reflect the specifics of the task?
- Criteria scope:
  - ☐ Is each individual criterion focused on a single quality dimension?

Evaluation Criteria and Human Experiments

- Human Experiment Instructions:
  - ☐ Are evaluation criteria defined in the experiment?
  - ☐ Are the criteria definitions precise?
  - ☐ Are the defined criteria independent?
  - ☐ Which human judgment elicitation method is used?
- Human Experiment Difficulty:
  - ☐ Is the evaluation task simple?

Evaluation Criteria and Metrics

- Targeted Language Level:
  - ☐ Does the metric measure the correct linguistic level?
- Task Specificity and Metrics:
  - ☐ Is the metric supposed to approximate the chosen evaluation criterion?

Data and Metrics

- Metric Assumptions:
  - ☐ Does the metric assume low quality of the outputs?
  - ☐ Is the metric supposed to work on corpus/segment level?
  - ☐ Has the metric migrated from another task?
- References:
  - ☐ Is the metric sensitive to the reference quality?
  - ☐ Is the number of references sufficient?

Metric Considerations

- ☐ Are the compared approaches typologically different?
- ☐ Is there any evidence that the metric correlates well with human judgements of the measured quality in the task at hand?

3.10 Chapter Summary

In this chapter we introduced the *Sanity Polygon Framework*, a conceptual framework for the design and analysis of NLG studies. Based on the review of prior NLG research, we argued that there is a need for further development of NLG methodology which can provide guidance during the design and analysis of an NLG experiment.

The framework consists of a set of recommendations grouped by relations between the common components of an NLG experiment. The described principles are not meant to exhaustively cover all interactions between the NLG components. In fact, even the number of the components might be revised in future — that is why we called the framework a "polygon", without specifying the number of the vertices in this geometrical analogy. For instance, we were considering including **approach** as one of the elements. However, eventually we decided against it, because the choice of a specific approach is almost predefined by the rest of the components.

We do believe that those principles are reasonable and supported by the analysis of the past NLG research. To facilitate an easier usage of the framework, we summarized the principles in a checklist form which we propose to apply for the analysis of an experiment at hand.

We believe that this framework can be applied to any NLP study, since the components are not NLG-specific. To provide evidence for this, in the subsequent chapters we will describe a series of NLG experiments that demonstrate the efficacy of the framework in approaching the research questions we outlined in the introduction chapter. The case studies were conducted in

three areas: data-to-text generation, sentence compression, and surface realization. The specific choice of these three tasks was influenced by several factors. First of all, all three are well-known text generation problems with established evaluation protocols, which means that our results could be easily compared to prior work. Secondly, they aligned well with our research agenda and the general trends in NLP: the encoder-decoder paradigm allows one to frame many NLP problems as an NLG task, and we used two contemporary shared tasks as a test field for our ideas. Finally, the chosen tasks exhibit a considerable application potential and are attractive from the practical perspective. Many companies are already relying on NLG techniques (Arria NLG[19], Narrative Science[20], Bloomberg[21]), which creates additional incentives for further development of NLG approaches. Our sentence compression experiments were largely motivated by the latter reason: most of that work was done during the internship which the author of this thesis took at Bloomberg L.P., London.

[19] https://www.arria.com
[20] https://narrativescience.com
[21] https://www.bloomberg.com

CHAPTER 4
Detecting Task Specification Issues

In the previous chapter we described the SPF formalized in a checklist form. In this chapter we approach *RQ1* and show how one can use SPF to detect and address possible issues in the relations between a task, experiment data and evaluation setup.

As an example task we are going to use the E2E NLG Challenge,[22] a 2017 shared task aimed at evaluation of state-of-the-art end-to-end data-driven NLG methods which became mainstream in modern NLG. Such methods jointly learn the text structure and surface realization patterns from non-aligned data. This significantly reduces the amount of human annotation effort needed for NLG corpus creation (Wen et al., 2015; Mei et al., 2016; Dušek and Jurčíček, 2016; Lampouras and Vlachos, 2016).

The main results from our participation in this task can be summarized as follows:

- We show how analyzing the task-data relation allows one to design more accurate end-to-end architectures.
- We question the claim that E2E NLG Challenge task specifications (Section 4.1) necessitate the use of data-driven approaches, and develop a simple template-based system which achieves performance comparable to state-of-the-art neural systems.
- We further analyze the final results and show how SPF reveals evaluation flaws which can cause misleading conclusions about the system performance.

In what follows we will first describe the task, the dataset and the baseline system provided by the organizers (Section 4.1). We take a closer look at the data set and analyze our observations in Section 4.2. Sections 4.3 and 4.4 introduce the two approaches we developed for the task. Section 4.5 describes evaluation results and the error analysis of the systems' predictions. Finally, we conclude with a discussion section which summarizes the obtained results (Section 4.6).

[22] `http://www.macs.hw.ac.uk/InteractionLab/E2E`

4.1 E2E NLG Challenge: Overview

The organizers of the E2E NLG Challenge shared a crowdsourced dataset of 50k instances in the restaurant domain (Novikova et al., 2017a). Each data instance consists of a dialogue act-based meaning representation (MR) and up to 16 references in natural language (Figure 4.1).

name[The Eagle]	*eatType[coffee shop]*
food[French]	*priceRange[moderate]*
customerRating[3/5]	*area[riverside]*
kidsFriendly[yes]	*near[Burger King]*

(a) Meaning Representation

The three star coffee shop, The Eagle, gives families a mid-priced dining experience featuring a variety of wines and cheeses. Find The Eagle near Burger King.

(b) Human Natural Language Reference

Figure 4.1: E2E NLG Challenge data specification: a meaning representation as input, a textual restaurant description as output.

The task was to generate an utterance from a given MR, which is both similar to human-generated reference texts and highly rated by humans. Text similarity was assessed using standard evaluation metrics: BLEU, NIST, METEOR, ROUGE-L, CIDEr. The final assessment was done via human ratings obtained using a mixture of crowdsourcing and expert annotations. Unfortunately, at the time of the competition the evaluation criteria were not defined (Section 3.6.2), so it was not even clear which metrics one should optimize to produce the optimal results.

E2E NLG Challenge dataset was split into training, validation and testing sets in a 76.5/8.5/15 proportion; it was also ensured that MRs in different sets are distinct. Development set inputs were paired with multiple references to facilitate more reliable model selection procedure; test data contained only input MRs. The objective was to develop and train an NLG system and submit restaurant description predictions for each of the test instances.

To facilitate a better assessment of the proposed approaches, the organizing team used TGen (Dušek and Jurčíček, 2016), one of the recent end-to-end data-driven systems, as a baseline. It is a seq2seq system with attention (Bahdanau et al., 2015). In addition to the standard seq2seq module, TGen uses beam search for decoding, incorporates a reranker over the top k outputs, penalizing the candidates that do not verbalize all attributes from the input MR. Some restaurant properties (such as restaurant names or location) appear in the dataset only a couple of times, which does not permit learning good vector representations of them. In order to circumvent this issue, TGen includes a delexicalization module which replaces the values with placeholder tokens when preprocessing the input data, and substitutes the placeholders with actual values as a post-processing step.

4.2 Data Analysis

As mentioned in Section 3.3, the relation between task and annotated data can reveal cues as to how to approach the task at hand.

We start with the analysis of the inputs. The meaning representations comprising the E2E NLG Challenge dataset used different modalities:

- Textual/logical MRs (Figure 4.2, left), comprise 80 % of the data: a list of comma-separated attribute-value pairs, where attribute values are shown in square brackets after each attribute. The order of attributes is randomised so that crowdworkers are not primed by the ordering used in the MRs.
- Pictorial MRs (Figure 4.2, right), the remaining 20 % of the data: semi-automatically generated pictures with a combination of icons corresponding to the individual attributes. The icons are located on a background showing a map of a city, thus allowing to represent the meaning of the attributes area and near.

1. *name[Loch Fyne],*
   eatType[restaurant],
   familyFriendly[yes],
   priceRange[cheap], food[Japanese]

2. *name[The Wrestlers],*
   familyFriendly[No], area[riverside],
   food[Italian], customerRating[5 of 5],
   priceRange[expensive],
   near[Cafe Adriatic],
   eatType[restaurant]

Figure 4.2: Examples of logical/textual (left) and pictorial (right) meaning representations. Figure taken from Dušek et al. (2020).

According to Dušek et al. (2020), using different modalities allowed for exploring the limits of the semantic complexity that crowdworkers can handle when using different types of restaurant descriptions. A pilot study conducted prior to the main annotation of the dataset concluded that using pictorial MRs allows to produce more natural, informative and diverse human references, but comes with two disadvantages: it is more expensive and introduces more noise (Novikova

et al., 2016). The specific 80/20 MR split mentioned above was fixed in order to keep noise and production costs low, while increasing diversity.

The fact that the dataset was crowdsourced and that the pictorial MRs allow producing more spontaneous descriptions alarmed us, since that would mean a potentially large portion of noise in the data (an issue we mentioned in Section 3.3.1). In order to better estimate the quality of the data and check if having multiple references per training instance mitigates this issue, we randomly sampled 100 training instances and manually checked their linguistic quality. Table 4.1 shows the most common errors we encountered.

Error type	Example	%
bad grammar	*it's French food falls within a high price range*	15
modified content	*area[riverside]* → *city centre*	12
dropped content	*priceRange[high]* → ∅	10
questionable lexicalization	*Adult-only Chinese restaurant, The Waterman, offers top-rated food in the city centre*	9
punctuation errors	*X is a coffee shop and also a Japanese restaurant great for family and close to Crowne Plaza Hotel*	6

Table 4.1: Data annotation error types found in 100 randomly sampled training instances.

Most mistakes come from ungrammatical constructions, e.g. incorrect phrase attachment decisions (*The price of the food is high and is located . . .*), incorrect usage of articles (*located in riverside*), repetitive constructions (*Cotto, an Indian coffee shop located in . . . , is an Indian coffee shop . . .*). Some restaurant descriptions follow a tweet-style narration pattern which is understandable, but ungrammatical (*The Golden Palace Italian riverside coffee shop price range moderate and customer rating 1 out of 5*).

A considerable number of instances have restaurant descriptions which contain information that does not entirely follow from the given input MR. These are cases in which input content elements are modified or dropped. The organizers mentioned that the crowdworkers were allowed to not verbalize certain fields (Novikova et al., 2017a). We suspect that this freedom could have left an opportunity for potential abuse by some annotators, which is why we view such cases as potentially harmful and include them as errors into the table.

A few instances (10 %) contained descriptions which we marked as questionable. They are grammatical, but are phrased in a way which we would rather avoid due to pragmatic and/or stylistic considerations. For example, restaurants which have *familyFriendly[no]* as part of the input MR are often described by crowdworkers as *adults-only* establishments, which has an undesirable connotation. Finally, it is necessary to mention that some crowdworkers followed inconsistent spelling and punctuation rules. The most prevalent cases of the former are those of hyphenating compound modifiers (*family friendly restaurant*, *the restaurant is family*

friendly), capitalizing MR attributes (*Riverside*, *Fast food*) and various typos (*neat* instead of *near*, *rage* instead of *range*). Punctuation errors were mainly restricted to missing a full stop at the end of a restaurant description or failing to delimit sentence clauses with commas.

Task-data checklist Based on our observations, we can fill in the task-data relation checklist as given below (see Section 3.9 for a general description of how to use the checklist).

Task and Data

- Data annotation:
  - ☑ Is the phenomenon at hand ambiguous? **Yes: it is possible to generate several semantically equal restaurant descriptions from the same pictorial/textual representation.**
  - ☑ Are the annotation guidelines simple? **Yes.**
  - ☑ Are the guidelines ambiguous? **Yes.**
  - ☑ Do annotators have the needed expertise? **No.**
  - ☑ How many annotators are there per data point? **8.27 on average.**
  - ☑ Does the data contain time-dependent phenomena? **No.**
- Data artifacts:
  - ☑ Is the data noisy? **Yes.**
  - ☑ Could the data be biased? **No.**
  - ☑ Is it a low-resource scenario? **No.**

The results of data analysis clearly show that a data-driven model would struggle to bypass the noise inherent to E2E NLG Challenge data annotations. If this is the case, one would need to propose a different approach to solve the task at hand.

In order to corroborate this hypothesis, we developed two very different approaches for the shared task. The first one (MODEL-D, standing for *data-driven*) fits the aim of the task to evaluate novel data-driven methods of data-to-text generation. It is an encoder-decoder neural system (Cho et al., 2014c; Sutskever et al., 2014) which is similar to TGEN, but uses a more efficient encoder module. The primary limitation of the baseline's architecture that we targeted is the sequential nature of the encoder. Given a set of MR key-value pairs, TGEN linearizes it into a sequence of tokens by concatenating keys and values. The resultant sequence is further fed to an RNN. RNNs have an advantage of being able to process variable-sized inputs. However, due to the auto-regressive nature of these networks, they also learn dependencies between sequence items,

which might not be desired in some cases (e.g. when encoding sets or data structures with non-linear structural relations). We decided to investigate ways of using the data properties in order to deal with encoding inputs of different sizes while refraining from imposing any dependencies between the constituting MR attributes, since each input MR is a set of items, not a sequence.

Section 4.4 introduces the second approach which is a simple template-based model (MODEL-T, standing for *template-based*). We viewed such a system as a necessary candidate for comparison, since the E2E NLG Challenge data was designed to learn models that produce *more natural, varied and less template-like system utterances* (Novikova et al., 2017a). We developed the system based on the results of the exploratory data analysis. The hypothesis was that the amount of noise present in the dataset would cause a data-driven model to output low-quality restaurant descriptions, while a template-based model would be resistant to the noise issues (Section 3.7.2).

4.3 Robust Encoding Strategy

MODEL-D was motivated by two important properties of the E2E NLG Challenge data:

- a fixed number of unique MR attributes;
- low diversity of the lexical instantiations of the MR attribute values.

Each input MR contained between three and eight unique attributes, which allowed us to associate a positional id with each attribute and omit the corresponding attribute names (or keys) from the encoding procedure. This shortened the encoded sequence, making the learning procedure easier for the encoder. This also unified the lengths of input MRs and thus allowed us to use simpler and more efficient neural networks which are not sequential and process input sequences in one step (e.g. multi-layer perceptron (MLP)).

One might argue that using an MLP would be complicated by the fact that neither the number of active (non-null value) input MR keys, nor the number of tokens constituting the corresponding values is fixed. For example, an MR key *price* may have a one-token value of *low* or a more lengthy *less than £10*. However, we analyzed training data realizations of the MR attribute values and found that they exhibited low variability: six out of eight keys had less than seven unique values, while the remaining two keys (*name*, *near*) denoted named entities and thus were easy to delexicalize. This allowed us to treat each value as a single token, even if it consisted of multiple words (e.g. *more than £30*, *fast food*).

Each predicted output was a textual description of a restaurant. Figure 4.3 shows a histogram of the distribution of the lengths of reference texts in the training data. A reference's length was measured as the number of tokens comprising the reference (including punctuation).[23] We used the value of 50 as a cut-off threshold, filtering out training instances with long restaurant descriptions.

[23] As reported by (Novikova et al., 2017a), the average number of words per reference in the E2E dataset is 20.1.

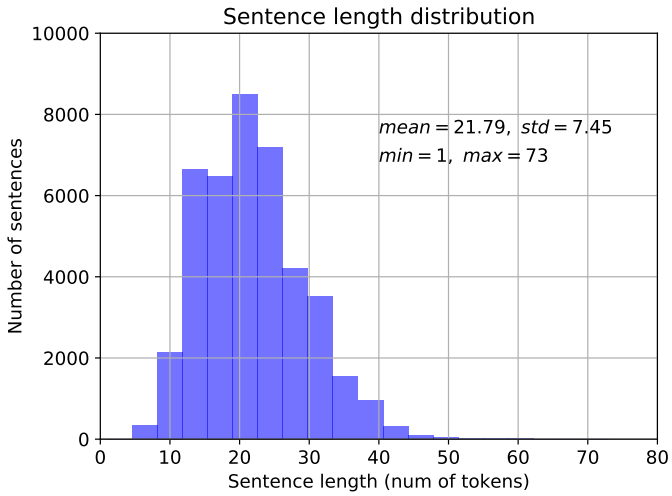


Figure 4.3: E2E NLG Challenge data analysis: length distribution of restaurant descriptions in the training data.

The overall architecture of the MODEL-D approach is shown in Figure 4.4. The system is an encoder-decoder model consisting of three main modules: an embedding matrix, one dense hidden layer as an encoder and an RNN-based decoder with GRU (Cho et al., 2014a).

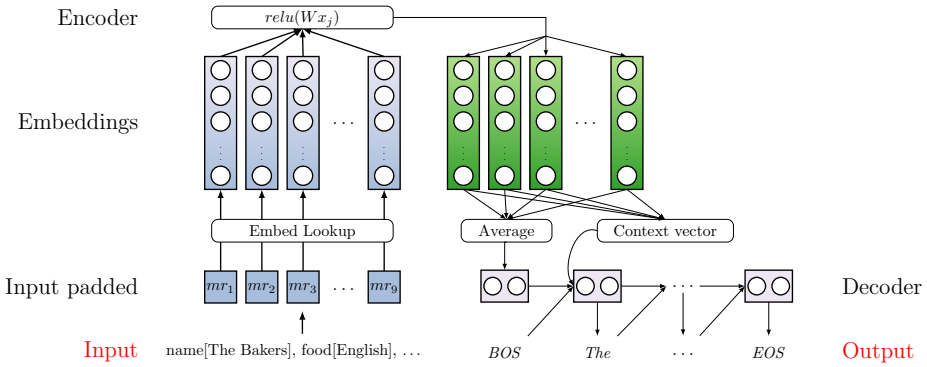


Figure 4.4: Schematic view of the neural network architecture (MODEL-D).

Let us first describe the input specifications of the model. We will use the following MR instance as a running example:

name[Wrestlers] *customerRating[high]*
familyFriendly[yes]

Considering the alphabetic ordering of the MR key names, we can assign positional ids to the keys as shown in Table 4.2. The remaining five keys are assigned dummy *PAD* values.

Given an instance of a *(MR, text)* pair, we decompose the MR into eight components (mr_j in Figure 4.4), each corresponding to a value for a unique MR key, and add an end-of-sequence

posID	*Key*	*Value*
1	*area*	*PAD*
2	*customerRating*	*high*
3	*eatType*	*PAD*
4	*familyFriendly*	*yes*
5	*food*	*PAD*
6	*name*	*Wrestlers*
7	*near*	*PAD*
8	*priceRange*	*PAD*

Table 4.2: Input representation of the running example, using positional ids.

symbol (*EOS*) to denote the end of the encoded sequence. For notation purposes, let us denote the total number of components as $N = 9$ (including *EOS*). Each component is represented as a d-dimensional embedding vector $x_j \in \mathbb{R}^d, j \in \{1, \ldots, N\}$. The embedding matrix which contains all such vectors is denoted as $E \in \mathbb{R}^{d \times |V|}$, where V is the vocabulary of unique tokens observed in the training data. Each embedding vector is further mapped to a dense hidden representation via an affine transformation followed by a ReLu function (Nair and Hinton, 2010):

$$h_j = relu(W x_j) \tag{4.1}$$

Here $W \in \mathbb{R}^{k \times d}$ is a weight matrix and $h_j \in \mathbb{R}^k$ is a dense representation of the MR component mr_j. We take an average of the encoder outputs and initialize the decoder with the resultant mean vector:

$$s_0 = \frac{1}{N} \sum_{j=1}^{N} h_j \tag{4.2}$$

Vectors h are further used by the decoder network, which in our case is a unidirectional GRU-based RNN with an attention module (Bahdanau et al., 2015). The decoder generates a sequence of tokens, one token at a time, until it predicts the *EOS* token. At timestep t the decoder defines a probability of generating a token y_t, based on the previously predicted word y_{t-1}, previous hidden state of the GRU s_{t-1} and the context vector c_t:

$$p(y_t | y_1, \ldots, y_{t-1}, x) = softmax(g(y_{t-1}, s_t, c_t)) \tag{4.3}$$

Here g is a transformation function that outputs a vocabulary-sized vector. The hidden state of the decoder is computed as follows:

$$s_t = gru(y_{t-1}, s_{t-1}, c_t) \tag{4.4}$$

The context vector c_t is a weighted sum of the input representations computed by the encoder:

$$c_t = \sum_{j=1}^{N} \alpha_{tj} h_j \tag{4.5}$$

Weights α_{tj} are computed as follows:

$$\alpha_{tj} = \frac{\exp(e_{tj})}{\sum_{l=1}^{N} \exp(e_{tl})}; \tag{4.6}$$

$$e_{tj} = v_a^T tanh(W_a s_{t-1} + U_a h_j) \tag{4.7}$$

Here $v_a \in \mathbb{R}^m$, $W_a \in \mathbb{R}^{m \times k}$, $U_a \in \mathbb{R}^{m \times k}$ are weight matrices storing the parameters of the attention module. Our model employs the greedy search decoding strategy and does not use any reranker module.

We implemented the network using the PyTorch deep learning library (Paszke et al., 2019). We used 256-dimensional randomly initialized embedding vectors, 512-dimensional MLP layer, and 512-dimensional uni-directional GRU decoder cells. We trained the system using stochastic gradient descent (Bottou, 1999), minimizing cross-entropy loss per predicted token, and used dropout of 0.1 after the embedding layer (Srivastava et al., 2014). We trained the system for 30 epochs with a learning rate of 0.1 and a batch size of 16, saving the predictions and performance scores at each epoch. After the training procedure was finished, we chose the model with the highest score, as measured on the development set.

The organizers proposed to evaluate the system outputs using five evaluation metrics. Note, however, that the evaluation criteria were not defined until the end of the competition, which hindered choosing the appropriate metric for optimization. Because it was not clear which one better reflects text quality, we decided to use a simple average of the metrics as a model selection criterion, which previously was found to be a sensible approach in such cases (Hastie and Belz, 2014). Before computing the average, the scores for each metric were normalized according to the following formula:

$$x_{norm} = \frac{x - min(x)}{max(x) - min(x)} \tag{4.8}$$

4.4 Template-based Approach

Taking into consideration the low lexical variation of the MR attribute values, one might be interested in whether it is possible to design a deterministic NLG system to tackle the task. We examined the ways MR attribute keys and values are verbalized in the training data and discovered that the majority of textual descriptions follow a similar ordering of MR attribute verbalizations:

```
[name] is a [familyFriendly] [eatType] which serves
[food] food in the [price] price range. It has a
[customerRating] customer rating. It is located in
the [area] area, near [near].
```

Here `[X]` denotes the value of the MR key X. This pattern became a central template of MODEL-T. Not all MR attribute verbalizations fit into this schema. First of all, some key-values might be missing (as in our running example from Section 4.3). Second, depending on the value,

this template verbalization should be adjusted. For example, if we were to use the template above, a key-value pair *customerRating[3 out of 5]* would be verbalized as *. . . has a 3 out of 5 customer rating*, which is not the best phrasing one can come up with. A better way to describe it is *. . . has a customer rating of 3 out of 5*. We incorporate such variations into MODEL-T with a set of simple rules which modify the general template depending on a specific value of an MR attribute.

As mentioned in Section 4.3, each instance's input can have up to eight MR attributes. In order to account for this fact, we decomposed the general template into smaller components, each corresponding to a specific MR attribute mentioned in the input. We further developed a set of rules which activate each component depending on whether an MR attribute is part of the input. For example, if *price* is not in the set of input MR attributes, then the general template becomes:

```
[name] is a [familyFriendly] [eatType] which serves
[food] food. It has a [customerRating] customer rating.
It is located in the [area] area, near [near].
```

Finally, we also added a simple post-processing step to handle specific punctuation and article choices.

When designing the templates, we observed that the provided data contains artifacts which cannot be attributed to the occasional noise or annotation guideline decisions made by the organizers. For example, 33 out of 5 996 (or 0.55 %) of training instances with the input attribute *food[Japanese]* had descriptions of Chinese restaurants as reference outputs. This probably was caused by incorrect interpretation of the input MR — as we mentioned in Section 4.1, the data originally was pictorial, so some crowdworkers might have chosen a wrong label for denoting the Japanese cuisine.

Another example is unmotivated specification of price ranges (*priceRange[cheap]* → *the cheap price of £10.50*) or customer ratings (*customerRating[high]* → *a 9 on a scale of 1-10*). These cases are most likely the result of some of the annotators' attempt to be creative, whereby they provided a subjective opinion on what constitutes *cheap* or *high* (Section 3.4.2).

4.5 Experiments

4.5.1 Automatic Metric Evaluation

Following the setup of the competition, we analyze the performance of the proposed approaches on the development set using five automatic metrics implemented in the provided scripts.[24]

All of the proposed metrics compute an n-gram overlap between the input and a set of references. The metrics include both corpus- and segment-level metrics. It is hard to say which metric should work best, since there is limited data on metric correlation with human judgments for the

[24] https://github.com/tuetschek/e2e-metrics

task at hand. The biggest weakness of the employed metrics is that they are reference-based, and, therefore, are sensitive to the quality of the gold standard.

Metric-related checklists Note that having several metrics for automatic evaluation makes it hard to use the checklist to perform an aggregate analysis on them. In some cases we cannot provide a definitive answer to a checklist question, because we have not found related information about a metric; sometimes the answer would vary, depending on the metric.

While the metrics employed in the shared task have been shown to suit the linguistic level being evaluated (Reiter and Belz, 2009), it is hard to say if the metrics can approximate the chosen evaluation criteria, because evaluation criteria were not defined until the beginning of human evaluation. This is problematic, because it makes it hard to decide how to combine the metrics for optimization.

Evaluation Criteria and Metrics

- Targeted Language Level:
  - ☑ Does the metric measure the correct linguistic level? **Yes: lexical and syntactic.**
- Task Specificity and Metrics:
  - ☑ Is the metric supposed to approximate the chosen evaluation criterion? **N/A: before human evaluation started, it was not clear what the criteria are.**

Data and Metrics

- Metric Assumptions:
  - ☑ Does the metric assume low quality of the outputs? **BLEU – yes, not enough evidence on the others.**
  - ☑ Is the metric supposed to work on corpus/segment level? **Both.**
  - ☑ Has the metric migrated from another task? **Yes: machine translation (BLEU, NIST, METEOR), summarization (ROUGE), image captioning (CIDEr).**
- References:
  - ☑ Is the metric sensitive to the reference quality? **Yes: see Section 3.7.2.**
  - ☑ Is the number of references sufficient? **Yes: see Section 3.7.2.**

Metric Considerations

- ☑ Are the compared approaches typologically different? **Yes: TGen and Model-D are data-driven, Model-T is template-based.**
- ☑ Is there any evidence that the metric correlates well with human judgements of the measured quality in the task at hand? **No.**

4.5.1.1 Results

The results of metric-based evaluation of the proposed approaches are shown in Table 4.3.

Metric	TGen	Model-D	Model-T
BLEU	0.6925	**0.7128** ± 0.013	0.6051
NIST	8.4781	**8.5020** ± 0.092	7.5257
CIDEr	2.3987	**2.4432** ± 0.088	1.6997
ROUGE-L	0.7257	**0.7378** ± 0.015	0.6890
METEOR	0.4703	**0.4770** ± 0.012	0.4678

Table 4.3: Evaluation results according to automatic metrics (development set).

In our comparison, we rely on the performance of TGen reported by the organizers of the shared task (Novikova et al., 2017a). Since we were provided with only one TGen prediction file and a single performance score, comparing score distributions is not possible and statistical significance tests are not meaningful due to the non-deterministic nature of the approaches based on neural networks and randomized training procedures (Reimers and Gurevych, 2017). In order to facilitate a fair comparison with other competing systems, we report the mean development score of Model-D (averaged across twenty runs with different random seeds) and performance variance for each automatic metric. The corresponding twenty predictions are available in our code repository.[25] Model-T is a deterministic system, so it is sufficient to report the results of a single run.

The results show that Model-D outperforms TGen as measured by all five metrics, albeit the performance variance is quite large. Model-T clearly scores below both TGen and Model-D. This is expected, since Model-T is not data-driven, and hence the texts it generates might be different from the reference outputs.

However, this does not yet mean that Model-D is better. First of all, we already mentioned that metric-based evaluation is just a proxy estimator of the quality of text candidates (Section 2.4.2). Previous studies have shown that widely used automatic metrics (including the ones employed in our competition) lack strong correlation with human judgments of various quality

[25] https://github.com/UKPLab/e2e-nlg-challenge-2017

facets (Scott and Moore, 2007; Reiter and Belz, 2009; Novikova et al., 2017b). Without clearly defined criteria it is hard to say if they make sense as good quality proxies (Section 3.6.1). They assess the content overlap between a reference text and the generated output, but they do not measure grammaticality or evaluate the discourse structure of the candidate output (Section 3.6.2). Moreover, in cases when a model ignores parts of the input when generating text or hallucinates content not given in the input, these metrics rely entirely on the quality of the references in their assessment of the candidates (Section 3.7.2). But since we already found that the data is noisy, it remains unclear whether high scores are even meaningful.

To make more solid conclusions, we performed manual error analysis of the predictions made by the compared systems. The results of the analysis are presented in the next section.

4.5.2 Error Analysis

We randomly sampled 100 input instances from the development set and retrieved the corresponding outputs from the official baseline prediction file provided by the organizers. After training MODEL-D we had twenty serialized model instances and prediction files. We randomly picked one instance and retrieved predictions for the sampled 100 inputs. Finally, we also extracted the corresponding predictions of our template-based model.

We already mentioned that prior to human evaluation, the shared task organizers did not define the evaluation criteria, which posed an additional challenge for system development and error analysis. We focused on generic errors, which make sense to look out for in many NLG scenarios. Table 4.4 shows the error types and number of mistakes found in each of the prediction files. The error types should be self-explanatory (sample predictions are given in Appendix A.2)

Error type	TGEN	MODEL-D	MODEL-T
dropped content	9	**49**	0
punctuation errors	1	**12**	0
modified content	4	4	0
bad grammar	4	1	0

Table 4.4: Common errors made by the compared models (100 randomly sampled development instances).

As far as the manual analysis goes, MODEL-T outputs descriptions with the best linguistic quality. Table 4.4 shows that the predictions of the template-based system contain no errors — this is because we incorporated our notion of semantic correctness and grammaticality into the templates' definition, which allowed MODEL-T to avoid the errors found in predictions of the other two approaches.

Note that although MODEL-T is able to produce grammatical restaurant descriptions, it inevitably suffers from low output variety. Its advantage lies in the fact that we can easily adjust it to generate more user-specific texts.

The majority of errors made by MODEL-D are either wrong verbalizations of the input MR values or punctuation mistakes. The latter ones are limited to the cases of missing a comma between clauses or not finishing a sentence with a full stop. An easy solution to this problem is adding a post-processing step which fixes punctuation mistakes before outputting the text. TGEN has an advantage here, since it uses a set of rules which correct punctuation errors, while MODEL-D is purely data-driven (and as we show in the following section, punctuation errors are common in the provided data).

A more interesting challenge is posed by those cases where our model drops or modifies some MR attribute values. According to the organizers, 40 % of the data by design contain either additional or omitted information on the output side (Novikova et al., 2017a): crowdworkers were allowed to not lexicalize attribute values which they deemed unimportant (Section 3.5.1). Taking this into consideration, we might conclude that MODEL-D outperforms TGEN, since MODEL-D generates texts which are more similar to the ones encountered in the training data. Unfortunately, the exact definition of importance used for the annotation is unknown to us, which is why we could not assess the validity of MODEL-D's behaviour in specific cases.

The results of manual data analysis show that MODEL-D indeed generates texts that are similar to the restaurant descriptions in the provided dataset. Unfortunately, using multiple references did not counter the noise present in some training instances (Section 3.3.2). One way of alleviating this problem could be reformulating the loss function to inform the system about the existence of multiple ways of generating a good restaurant description. Given a training instance, MODEL-D would generate a corresponding candidate text which could be compared to all human references. Each comparison results in computing a certain cost; the gradients could be then computed on the minimal cost among all comparisons. The approach could be further improved by adding a post-processing step which fixes punctuation and/or occasional spelling errors.

4.5.3 Final Evaluation

Following the shared task requirements, we had to restrict ourselves to submitting the restaurant descriptions produced by only one system. Even though E2E NLG Challenge was focusing more on data-driven approaches, for the final submission we have chosen MODEL-T's outputs: despite lower metric scores, they contained more grammatical texts and kept all input information in the generated descriptions.

Metric evaluation The results of the final automatic metric evaluation on the test data are presented in Table 4.5. None of the participating systems outperformed the rest according to all metrics. To put the scores obtained by MODEL-T into some perspective, we include the highest reported scores among all the participants (rightmost column). These scores were obtained by various RNN-based seq2seq approaches with attention. Note, however, that most of the top-scoring systems relied on ensembling seq2seq systems (Gehrmann et al., 2018), augmenting training data (Oraby et al., 2018; Tandon et al., 2018), or doing both (Juraska et al., 2018). The full comparison of the systems can be found in (Dušek et al., 2018).

Metric	MODEL-T	Best result
BLEU	0.5657	0.6619
NIST	7.4544	8.6130
METEOR	0.4529	0.4529
ROUGE-L	0.6614	0.7083
CIDEr	1.8206	2.2721

Table 4.5: Final automatic metric evaluation results on the E2E NLG Challenge test set.

Human evaluation Once the final metric evaluation results were known, the organizers shared the details of human evaluation as well. Those experiments relied on the following two quality criteria:

- **Quality**: *an overall quality of the utterance, in terms of its grammatical correctness, fluency, adequacy and other important factors*.
- **Naturalness**: *the extent to which the utterance could have been produced by a native speaker*.

The final evaluation results were produced by the TrueSkill™ algorithm (Herbrich et al., 2006) which has been shown to reduce the amount of collected human evaluation data without compromising the final ranking results (Novikova et al., 2018).

In total, the task considered 20 primary submissions, which is a rather large number to compare systems directly. TrueSkill™ allows to perform fewer direct comparisons between systems to establish their overall ranking. The algorithm gradually updates a Bayesian estimate of each system's accuracy according to the *surprisal* of pairwise comparisons of individual system outputs. Then it creates a partial ordering into significance clusters established by bootstrap resampling (Sakaguchi et al., 2014). For the shared task, the algorithm was run 200 times, which produced different rankings each time as pairs of system outputs for comparison were randomly sampled. This was done in order to determine the range of ranks where each system is placed 95 % of the time or more often. Five clusters were formed of systems whose rank ranges overlap, thus producing system rankings. Although the numerical scores produced by the algorithm are not directly interpretable, the relative ranking of a system in terms of its range and cluster is important: in terms of performance, systems within one cluster are considered to be equal.

The final human evaluation ranking results are presented in Figure 4.5. MODEL-T (denoted as ♠**TUDA** in the figure) was assigned to the second best cluster (out of five) both in terms of quality and naturalness, despite the much lower metric scores.

Criteria-related checklists The analysis of the human evaluation setup using SPF reveals several points of concern. First of all, the quality criteria were not defined precisely (Section 3.5.1). We already mentioned that using *overall quality* as a criterion makes human assessors conflate various quality dimensions (Section 3.4). Secondly, the definitions of the evaluation criteria, used in the task, make it difficult to distinguish them, because:

Quality

#	TrueSkill	Rank	System
1	0.300	1–1	♡ SLUG
	0.228	2–4	♠ TUDA
	0.213	2–5	♡ GONG
	0.184	3–5	♣ DANGNT
	0.184	3–6	♡ TGEN
	0.136	5–7	♡ SLUG-ALT *(late)*
	0.117	6–8	♢ ZHAW2
2	0.084	7–10	♡ TNT1
	0.065	8–10	♡ TNT2
	0.048	8–12	♡ NLE
	0.018	10–13	♢ ZHAW1
	0.014	10–14	♣ FORGE1
	-0.012	11–14	♢ SHEFF1
	-0.012	11–14	♡ HARV
3	-0.078	15–16	♠ TR2
	-0.083	15–16	♠ FORGE3
	-0.152	17–19	♡ ADAPT
4	-0.185	17–19	♡ TR1
	-0.186	17–19	♡ ZHANG
5	-0.426	20–21	♡ CHEN
	-0.457	20–21	♡ SHEFF2

Naturalness

#	TrueSkill	Rank	System
1	0.211	1–1	♡ SHEFF2
	0.171	2–3	♡ SLUG
	0.154	2–4	♡ CHEN
	0.126	3–6	♡ HARV
	0.105	4–8	♡ NLE
	0.101	4–8	♡ TGEN
2	0.091	5–8	♣ DANGNT
	0.077	5–10	♠ TUDA
	0.060	7–11	♡ TNT2
	0.046	9–12	♡ GONG
	0.027	9–12	♡ TNT1
	0.027	10–12	♡ ZHANG
	-0.053	13–16	♡ TR1
	-0.073	13–17	♡ SLUG-ALT *(late)*
3	-0.077	13–17	♢ SHEFF1
	-0.083	13–17	♢ ZHAW2
	-0.104	15–17	♢ ZHAW1
4	-0.144	18–19	♣ FORGE1
	-0.164	18–19	♡ ADAPT
5	-0.243	20–21	♠ TR2
	-0.255	20–21	♠ FORGE3

Figure 4.5: Final human evaluation results: measurements of *quality* (left) and *naturalness* (right) as computed by the TrueSkill™ algorithm. The first column denotes the significance cluster number, followed by the TrueSkill™ value, range of ranks where the system falls in 95% of cases or more and, finally, system name. Significance clusters are separated by a dotted line. System architectures are colour-coded: seq2seq, other data-driven, rule-based, template-based. Figure taken from Dušek et al. (2018).

- A native speaker is unlikely to produce a low-quality utterance.
- A high-quality text is very likely to get a high naturalness score.

In such a setup, there is a high chance that human assessments of the two dimensions would correlate, rendering one of the criteria redundant (Novikova et al., 2017b). The reason why this did not happen in the E2E NLG Challenge evaluation was probably because human judgements were collected in two separate steps with a different setup: when collecting *quality* ratings, system outputs were presented to crowd workers together *with* the corresponding meaning representation. However, when collecting *naturalness* ratings, system outputs were presented to crowd workers *without* the corresponding meaning representation (Dušek et al., 2018).

Finally, *semantic correctness*, which we would expect to be an important quality dimension in the task of restaurant description generation, was not among the evaluation criteria. The organizers omitted it, since the training instances of E2E NLG Challenge do not always verbalise all MR attributes (Dušek et al., 2020). This raises further questions about the evaluation protocol, because all the employed automatic metrics measure content overlap between references and system outputs, i.e. they are more suited to assess the semantic correctness of the outputs, rather than

naturalness, for example. In other words, there is a clear mismatch between the automatic metrics and evaluation criteria used in the human evaluation experiments Section 3.4.

Based on the analysis above, we can fill in the SPF checklists as follows:

Task and Evaluation Criteria

- Task-Criteria Match:
  - ☑ Do the criteria reflect the specifics of the task? **No: semantic correctness was not included.**
- Criteria scope:
  - ☑ Is each individual criterion focused on a single quality dimension? **No: the criteria are high-level and abstract.**

Evaluation Criteria and Human Experiments

- Human Experiment Instructions:
  - ☑ Are evaluation criteria defined in the experiment? **Yes.**
  - ☑ Are the criteria definitions precise? **No, especially the *quality* criterion.**
  - ☑ Are the defined criteria independent? **No.**
  - ☑ Which human judgment elicitation method is used? **Preference-based.**
- Human Experiment Difficulty:
  - ☑ Is the evaluation task simple? **Yes.**

4.6 Discussion

The shared task allowed us to test the principles underlying the SPF. It helped us determine system output requirements and choose the right approach for submission. There are several important observations and conclusions we made.

4.6.1 Task Requirements and Evaluation Criteria

Metric evaluation results gave us an impression that MODEL-D is much better than both TGEN and MODEL-T. However, manual inspection revealed that both systems have their strong and weak sides. The template-based system produces the most grammatical results, but suffers from

low output variety. MODEL-D and TGEN generate more variable outputs which, nevertheless, occasionally contain grammatical errors. Both TGEN and MODEL-T try to verbalize every input MR value, but the organizers expected the systems to perform some content filtering. So, which model should have been chosen to solve the task at hand?

Different NLG tasks have different requirements. For example, low diversity of the generated candidates could be an issue for a chat-bot application, but an information retrieval system would be oblivious to that. We viewed the task of generating restaurant descriptions as a purely information-seeking real-world scenario. Note that in the research paper that introduced the E2E NLG Challenge dataset (Novikova et al., 2017a), diversity of the restaurant descriptions was presented as one of the main challenges for the data-driven approaches to tackle. Yet, it was not mentioned as one of the desired properties of the system outputs, none of the employed metrics could be used to capture that quality facet, and it was not one of the criteria in the human evaluation experiments either. This made us ignore the variability criterion and follow the generic NLG requirement of grammaticality and semantic correctness and submit MODEL-T's predictions which produced grammatical outputs while keeping all input information in the generated text. Judging from the evaluation results, our guess was correct, since the simple template-based system achieved high human assessment scores.

4.6.2 Development Costs vs. Quality Trade-off

This point is related to the previous one. We acknowledge the importance of developing data-driven models for solving complex problems. However, considering the trade-off between system-building cost and output quality, we would still choose a simple template-based model.

We spent roughly three hours designing and debugging the template model. As a result, it gives consistent, grammatical output, which can be further easily tailored to a particular user by adjusting the templates' content. On the other hand, MODEL-D took us approximately a month to develop and several days to train. Yet, both models generated texts of comparable linguistic quality, but the latter approach suffered from semantic errors.

The E2E NLG Challenge focuses on end-to-end data-driven NLG methods, which is why systems like MODEL-T might not exactly fit into the task setup. Nevertheless, we hope that our observations and findings facilitate a better understanding of the advantages and disadvantages of various NLG approaches. Ultimately, a task can be approached more efficiently if its design focuses on the properties of the targeted research problem.

4.6.3 Crowdsourcing and Business Sensitivity

How can one determine if a model makes unreasonable predictions due to the noise in the training data it learned from, or due to some bad system design decisions? Unlike the latter, the former is easy to spot. Substantial improvements to the output quality of a system could be gained by addressing the issue of data noisiness.

Consider the following hypothetical prediction candidates:

- *The Bakers is a restaurant serving English food.*
- *The Bakers is a restaurant at the riverside, near The Wrestlers.*

The two predictions are complementary in terms of their *content* and are equal in terms of *grammaticality*. Which one is better? The first sentence mentions the type of the food served, but omits the restaurant's location; the second candidate does the opposite. However, both outputs would probably be rated equally by human assessors in the current task setup, even though we do not know which content could be dropped and which should be kept intact.

Here the problem we are concerned about is not the question whether to separate content selection from surface realization or not. The issue is that the optimal output of an NLG system is context-sensitive. The task at hand is generating restaurant descriptions, which implies a certain degree of domain-specific *business sensitivity* which not all crowdworkers are concerned about. A user looking for family-friendly restaurants might be interested in family-friendly restaurants only. If a restaurant recommendation application decides to omit this information, the user will be very unsatisfied, which has direct implications in real-world business (Levin et al., 2017).

4.7 Chapter Summary

In this chapter we were aiming to address *RQ1* and *RQ2* by means of applying SPF to an NLG task and showing how one can use the framework to detect and address possible issues in the relations between a task, experiment data and evaluation setup. We described the results of our participation in the E2E NLG Challenge, which we viewed as a case study to provide empirical evidence of the utility of SPF.

We first analyzed the data which the organizers provided us with. We found that the lack of annotators' expertise and the excessive freedom given to the annotators made the data annotations noisy. We hypothesized that a data-driven model would struggle to bypass this issue.

Bearing this in mind, we developed two conceptually different approaches. First, we designed a more efficient data-driven system by exploiting the patterns in the available data. Second, we proposed a different design strategy which is based on the analysis of the task specifications, and developed a template-based approach to the task.

We further evaluated our approaches. Using SPF to analyze metric-based evaluation was complicated by the fact that the shared task used several of them. Nevertheless, we managed to obtain some insights into the assessment results. The main observation was that the chosen metrics are sensitive to the reference quality, and, given the noise in the data, should produce misleading scores.

We followed metric evaluation with manual error analysis and showed that our data-driven system indeed generated noisy texts that are similar to the restaurant descriptions in the provided dataset. In this way we showed how error analysis connects prediction problems with the results of data analysis.

For the final submission we chose the predictions from the template-based system. Human evaluation results proved the advantage of our strategy: our simple approach was in the second best cluster of systems (out of five).

Lastly, we explained how questionable design decisions may lead to negative repercussions at later evaluation stages. We also connected the task specifications to a real-world scenario and hinted on the potential issues one might face in the case of a real-world application.

In the next chapter we will show how SPF can be used to critically analyze evaluation setups, revealing weak spots of system assessment protocols.

CHAPTER 5
Detecting Evaluation Discrepancies

In the previous chapter we showed how one can apply SPF to perform critical data analysis and detect potential task-data relation issues. We further showed how the obtained results can be used to develop an approach that closely follows task specifications and performs well in practice, despite the pessimistic results of metric-based evaluation. In this chapter we would like to describe another case study in which we used the SPF principles to detect evaluation discrepancies, showcasing it in the task of sentence compression.

The study describes our attempt to improve the current state-of-the-art techniques of sentence compression. As a result, we show that sometimes neither metric, nor conventional human evaluation is sufficient to draw conclusions about system performance. We demonstrate how a system can fit the data to achieve state-of-the-art metric scores, while falling short in terms of actual output quality. In contrast to the results reported in previous work that showed correlation between human judgements of the system outputs and their metric scores (Filippova et al., 2015; Wang et al., 2017), our manual analysis of a state-of-the-art system outputs demonstrated that high metric scores may only indicate a better fit to the data, but not necessarily better outputs, as perceived by humans.

5.1 Sentence Compression: Task Overview

Sentence compression is an NLP task in which a system produces a concise, grammatical summary of a given sentence, while preserving the *important content of the original input sentence*. Both abstractive (Cohn and Lapata, 2008; Rush et al., 2015) and extractive (Filippova and Altun, 2013; Filippova et al., 2015; Wang et al., 2017; Zhao et al., 2018) approaches have been proposed to tackle this problem. Following most researchers, we focus on the extractive methods and treat it as a deletion-based task where each compression is a subsequence of tokens from its original sentence. An example sentence and two possible compressions are shown in Figure 5.1.

Dickinson, who competed in triple jump at the 1936 Berlin Games, was also a bronze medalist in both the long jump and triple jump at the 1938 Empire Games.

(a) Input sentence

Dickinson competed in triple jump at the 1936 Berlin Games.

(b) Compression A

Dickinson was a bronze medalist in the long jump and triple jump at the 1938 Empire Games.

(c) Compression B

Figure 5.1: Sentence compression example: an input sentence and two possible compression candidates.

5.2 Data Analysis

In our experiments we used the most common benchmark dataset for sentence compression, the so-called *Google Dataset*, which was introduced by Filippova and Altun (2013).[26] Prior work has shown that the headline and the first sentence of a news article are often semantically similar (Dorr et al., 2003). Following this reasoning, Filippova and Altun (2013) collected a corpus of news articles in English from the Internet, and extracted the headline (h) and the first sentence (s_1) from every article. The authors noted that headlines rarely constitute extractive compressions of the first sentences. On top of that, headlines in general have very specific stylistic features which sometimes violate language grammar. For example, a headline may omit certain words and appear incomplete. This ultimately makes it hard for a supervised deletion-based system to learn useful compression patterns.

Coal company Massey could have corporate charter revoked

(a) Article headline (h)

Massey Energy, the fourth largest coal company in the country, could have its corporate charter revoked if public interest organizations have their way.

(b) The first sentence of the article (s_1)

Massey Energy, the coal company, could have its corporate charter revoked.

(c) Extractive compression of the first sentence of the article ($\hat{s}_1$)

Figure 5.2: The Google Dataset: an example of a headline, the first sentence of the corresponding news article, and an extractive compression of the first sentence.

[26] https://bit.ly/2ZvTK9z

The authors therefore proposed to use the original headline h and apply a set of heuristics to it, to produce an extractive compression ($\hat{s}_1$) of the first sentence s_1. The suggested corpus creation procedure thus resulted in a collection of pairs (s_1, $\hat{s}_1$) which correspond to the input and expected output of a sentence compression system (Figure 5.2).

The resulting dataset contains 200 000 training and 10 000 evaluation instances; the first 1 000 data points from the latter are commonly used as a test set and the remaining 9 000 as a development set.

Quality As mentioned above, the dataset was created automatically, hence no annotation guidelines were developed. Consequently, the notion of importance for keeping certain tokens and pruning others, was not precisely defined either. On top of that, the heuristic rules used during the corpus creation could produce erroneous outputs. For these reasons, one could expect more inconsistent or noisy annotations than is usually observed in human-annotated data. To estimate the quality of the corpus, Filippova and Altun (2013) randomly selected 50 sentence-compression pairs from the dataset and conducted a human evaluation experiment.

The human raters were supposed to assess data quality in terms of *readability* and *informativeness*. Note, however, that the authors defined the former as "also known as grammaticality and fluency", and the latter as "also known as importance and representativeness". We view this decision as rather questionable, since it means conflating several evaluation criteria into one and a potential danger of human annotators mixing various criteria during the experiment (Section 3.5.1).

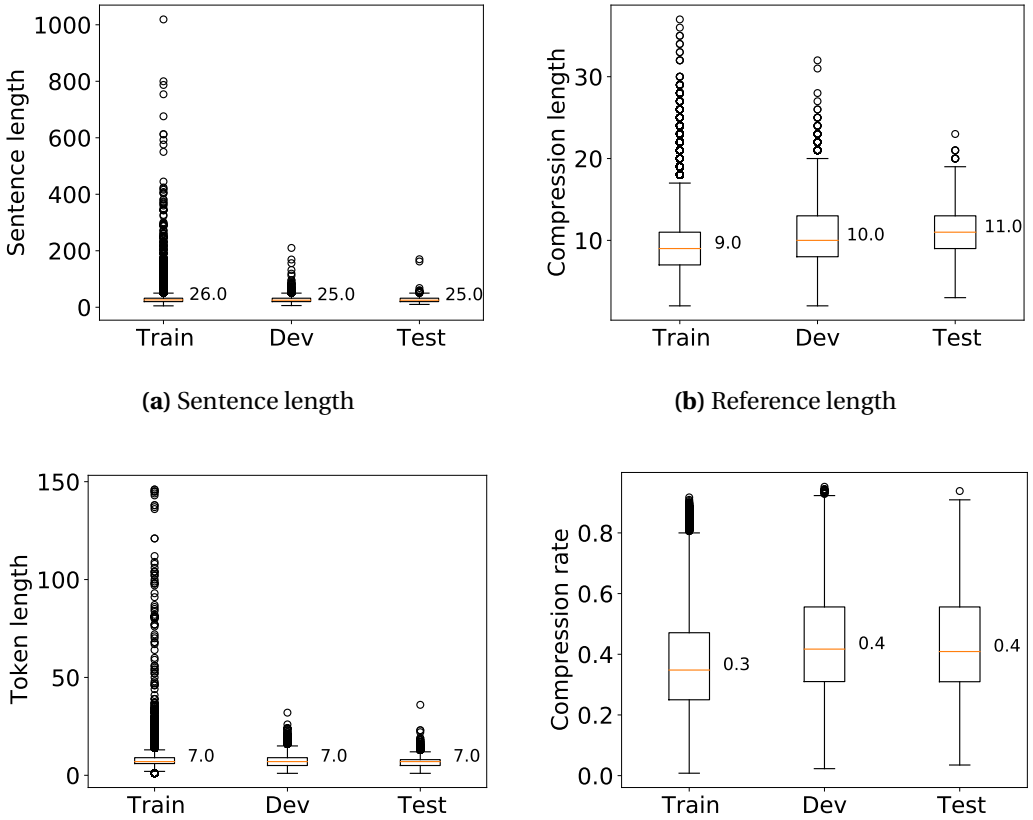
Three raters were asked to rate the sentence-compression pairs on a 5-point numerical rating scale. As compressions, the raters were given either the original news headlines, or the heuristically created extractive compressions. Table 5.1 shows the results of the study, which were rather reassuring, because quality-wise the extracted headlines seemed comparable with human-written ones.

	AVG read	AVG info
Headline (h)	4.36 (0.75)	3.86 (0.79)
Compression ($\hat{s}_1$)	4.26 (1.01)	3.70 (1.04)

Table 5.1: Results of the human experiment assessing the quality of the Google Dataset. The scores denote average *readability* and *informativeness*, with standard deviation in brackets. Figure adapted from Filippova and Altun (2013).

Filtering Our exploratory data analysis showed that the distribution of the corpus data is skewed (Figure 5.3): the data contains very long sentences and tokens, while the median values are rather small. This is not surprising though, given the nature of the data: it was automatically constructed and could be rather noisy.

In order to remove outliers and fit the computation budget, we removed instances which contained sentences longer than 50 tokens and compressions longer than 17 tokens. We also removed examples with tokens longer than 15 characters, since those in most cases denoted



(a) Sentence length

(b) Reference length

(c) Token length

(d) Compression ratio values

Figure 5.3: Data analysis of the Google Dataset: length distributions of sentences, ground-truth compressions and tokens, and the distribution of compression ratio values. The numbers to the right of each box denote median values.

website links. Finally, we excluded cases with a compression ratio of more than 0.85 — those rare cases in most cases were too long to qualify as compressions. Evaluation on the development and test sets was done without any data filtering.

Task-data checklist Below is the task-data relation checklist which we filled based on our observations. Note that we can't answer the questions pertaining data annotation, since the corpus was constructed automatically.

At the time of writing, the Google Dataset can be considered as a rather old one (published in 2013). Given that news articles report about contemporary issues, the corpus is likely to describe time-dependent phenomena. However, in our study, we do not expect this to affect the results of the sentence compression system's evaluation, because the system is evaluated on the test portion of the same dataset, which belongs to the same time period as the training data. For this reason,

we marked the related checklist question in black. Same reasoning is applied to the question about data bias.

Task and Data

- Data annotation:
  - ☑ Is the phenomenon at hand ambiguous? **Yes: several compressions might be valid, but there is only one reference.**
  - ☑ Are the annotation guidelines simple? **N/A: corpus was automatically constructed.**
  - ☑ Are the guidelines ambiguous? **N/A: same as above.**
  - ☑ Do annotators have the needed expertise? **N/A: same as above.**
  - ☑ How many annotators are there per data point? **N/A: same as above.**
  - ☑ Does the data contain time-dependent phenomena? **Yes.**
- Data artifacts:
  - ☑ Is the data noisy? **Yes.**
  - ☑ Could the data be biased? **Yes.**
  - ☑ Is it a low-resource scenario? **No.**

5.3 BERT-based Sentence Compression

Most modern deletion-based compression systems adopt either a tree-pruning, or a sequence labeling approach. The former uses syntactic information to navigate over a syntactic tree of a sentence and decide which parts of it to remove (Knight and Marcu, 2000; McDonald, 2006; Filippova and Altun, 2013). With the advent of seq2seq models it became possible to skip the syntactic parsing step and solve the task directly, by processing a sentence one token at a time and making binary decisions as to whether to keep a token or delete it (Filippova et al., 2015; Wang et al., 2017; Zhao et al., 2018; Kamigaito and Okumura, 2020). The advantages of such approaches include a smaller chance of introducing error propagation from incorrect parsing decisions, as well as higher training and inference speed.

For a long time, the space of seq2seq models has been dominated by different variants of RNN. However, a more recent Transformer architecture (Vaswani et al., 2017) has shown very promising results in many NLP tasks. Given the success of Bidirectional Encoder Representations from Transformers (BERT) (Devlin et al., 2019), and the fact that there has been no empirical

evaluation of its performance in sentence compression, we fill this gap and find out how well BERT-based models can cope with the task.

BERTUNI A schematic view of the proposed architecture is given in Figure 5.4. We used pre-trained BERT-base-uncased model weights[27] provided by the HuggingFace library (Wolf et al., 2020), and implemented a simple BERTUNI model which encodes the source sentence $S = \{w_1, w_2, \dots w_n\}$ and produces a sequence of vectors $V = \{v_1, v_2, \dots v_n\}, v_i \in \mathbb{R}^h$. Each vector is fed into a feedforward neural network (FFNN) with a logistic function as a non-linear function to produce a score $s_i \in [0, 1]$. If $s_i \geq 0.5$, the model output is 1 (and 0, otherwise).

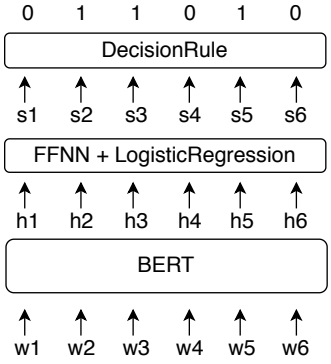


Figure 5.4: Sentence compression experiments: BERTUNI architecture.

A simple decision rule was used to make a binary prediction:

$$decision = \begin{cases} 1, & \text{if } s \geq 0.5, \\ 0, & \text{otherwise.} \end{cases}$$

The network components were implemented in PyTorch. We followed Devlin et al. (2019) and used the same hyperparameters for the BERT module; the additional dense layer is implemented as a 768-dimensional feedforward neural network. We trained the system for 30 epochs with a batch size of 128, using the Adam optimizer (Kingma and Ba, 2015) with a learning rate of 0.001, minimizing the binary cross-entropy loss per predicted label. After the training procedure was finished, we chose the model with the lowest loss, as measured on the development set.

5.4 Experiments

5.4.1 Automatic Metric Evaluation

For automatic evaluation of sentence compression systems, most researchers follow Filippova et al. (2015) and use two metrics. The first one is F1-score, the harmonic mean of the precision (P) and recall (R) in terms of tokens kept in the target ($\hat{s}_1$) and the output compressions (o):

[27] `https://huggingface.co/bert-base-uncased`

$$F1(o|\hat{s}_1) = 2 \cdot \frac{P(o|\hat{s}_1) \cdot R(o|\hat{s}_1)}{P(o|\hat{s}_1) + R(o|\hat{s}_1)} \qquad P(o|\hat{s}_1) = \frac{|o \cap \hat{s}_1|}{|o|} \qquad R(o|\hat{s}_1) = \frac{|o \cap \hat{s}_1|}{|\hat{s}_1|} \tag{5.1}$$

The second metric is compression ratio (CR) which is calculated as the length of the output compression in tokens, divided over the length of the original sentence:

$$CR(o|s_1) = \frac{len(o)}{len(s_1)} \tag{5.2}$$

The first metric is supposed to show how close the model outputs are to the references. The latter one is supposed to measure how the compression's conciseness.

To make our results comparable to previous work, in our experiments we followed the same convention. However, there are two important issues we would like to comment on. First, data-driven sentence compressors are likely to produce outputs with a compression ratio most commonly seen in the training data. In other words, CR becomes less a property of a system and more a characteristic of the dataset. This is supported by the fact that most approaches evaluated on the same dataset have similar compression ratios (in the case of the Google Dataset, it is in the range of 0.38–0.43, see Table 5.2).

Secondly, it is not entirely clear how to treat compression ratio values: is a CR of 0.4 better than a CR of 0.5? Intuitively, yes, because it means a more concise compression. However, the compression is really better only if it retained more valuable information from the source. On the other side, defining the notion of informativeness/importance in sentence compression (and document summarization, in general) is an open research problem (Peyrard, 2019). This means that the CR metric is a one-sided proxy, too crude to be used for real automatic evaluation without balancing it with some recall-oriented metric. In other words, this metric does not entirely satisfy the *task-specificity* principle of SPF (Section 3.6.2). This issue has been discussed by Napoles et al. (2011) who advocated for comparing systems with similar compression ratios.

Another important point to note is that the corpus has only one reference compression per sentence. This is not ideal, since there might be several valid ways to construct a compression from a sentence. An example was given at the beginning of this chapter (Figure 5.1). *Compression A* is the actual reference for the given sentence in the Google Dataset. *Compression B* is another plausible compression, but with the established evaluation metrics this compression would score very low because of the insignificant token overlap with the reference.

Metric-related checklists Taking this into consideration, we fill in the metric-related checklists as follows:

Task and Evaluation Criteria

- Task-Criteria Match:

- ☑ Do the criteria reflect the specifics of the task? **Yes: F1-score approximates a measure of informativeness, while compression ratio measures the conciseness of the output compression.**

- Criteria scope:
  - ☑ Is each individual criterion focused on a single quality dimension? **No: the tokens retained in the compression are chosen based on the notion of importance (see related work in Section 5.1). However, *importance* is a generic, high-level term.**

Evaluation Criteria and Metrics

- Targeted Language Level:
  - ☑ Does the metric measure the correct linguistic level? **Yes.**
- Task Specificity and Metrics:
  - ☑ Is the metric supposed to approximate the chosen evaluation criterion? **Yes.**

Data and Metrics

- Metric Assumptions:
  - ☑ Does the metric assume a low quality of the outputs? **No (we have not found any evidence of that in the literature).**
  - ☑ Is the metric supposed to work on corpus/segment level? **Segment.**
  - ☑ Has the metric migrated from another task? **No.**
- References:
  - ☑ Is the metric sensitive to the reference quality? **Yes: it measures word overlap between system outputs and reference compressions.**
  - ☑ Is the number of references sufficient? **No.**

Metric Considerations

- ☑ Are the compared approaches typologically different? **No: see Section 5.4.1.1.**
- ☑ Is there any evidence that the metric correlates well with human judgements of the measured quality in the task at hand? **Yes: see Napoles et al. (2011) and Filippova et al. (2015).**

5.4.1.1 Compared Approaches

To put the evaluation of our approach into better context, we compare it with the following systems. All systems predict a sequence of binary labels which decide which tokens to keep or remove from the input sentence.

LSTM Filippova et al. (2015) uses a three-layer uni-directional LSTM network (Hochreiter and Schmidhuber, 1997) and pretrained WORD2VEC (Mikolov et al., 2013) embeddings as input representations. For comparison, we use the results for LSTM-PAR-PRES, the best configuration reported in (Filippova et al., 2015). This system parses the input sentence into a dependency tree, encodes the tree structure and passes the aggregated feature representations to the decoder LSTM. Unlike our approach, this system relies on beam search at inference time.

BiLSTM BILSTM (Wang et al., 2017) builds upon LSTM approach, but introduces several modifications. It employs a bi-directional LSTM encoder and enriches the feature representation with syntactic context. In addition, it uses integer linear programming (ILP) methods to enforce explicit constraints on the syntactic structure and sentence length of the output.

Evaluator-LM EVALUATOR-LM (Zhao et al., 2018) uses a bi-directional RNN to encode the input sentence and predict a binary label for each input token. In addition to token embeddings, the network uses vector representations of POS tags and dependency relations. The system is trained using the REINFORCE algorithm (Williams, 1992), the reward signal comes from a pretrained syntax-based language model (LM).

SLAHAN SLAHAN (Kamigaito and Okumura, 2020) is a modular seq2seq model that consists of several components. The system encodes a sequence of tokens using a combination of pretrained embeddings (Glove (Pennington et al., 2014), ELMO (Peters et al., 2018), BERT) and parses the input into a dependency graph. Three attention modules are employed to encode the relations in the graph, their weighted sum is passed to a selective gate. The output of the latter forms an input to a LSTM decoder.

5.4.1.2 Results

Table 5.2 shows that, despite its simplicity, the proposed approach achieved very competitive scores.

We mentioned previously that comparing single performance scores (and not score distributions) of neural approaches is questionable, because training neural models is non-deterministic in many aspects and depends on random weight initialization, random shuffling of the training data for each epoch, applying random dropout masks (Reimers and Gurevych, 2017). This makes it hard to compare the scores reported in previous works and our approach. To facilitate a fair comparison with future systems, we report the mean and standard deviation of the BERTUNI scores averaged across ten runs with different random seeds.

Model	F1↑	CR↓
EVALUATOR-LM	0.851	0.39
BILSTM	0.800	0.43
LSTM	0.820	**0.38**
SLAHAN	0.855	0.407
BERTUNI	**0.857** ±0.002	0.413 ±0.004
BERTUNI (dev)	**0.860** ±0.001	0.418 ±0.004

Table 5.2: The performance of the BERTUNI model on the test portion of the Google Dataset, compared to recent approaches. The last row shows BERTUNI's performance on the development set.

In order to understand where BERTUNI fails and what we could potentially improve upon, we conducted manual error analysis of its predictions.

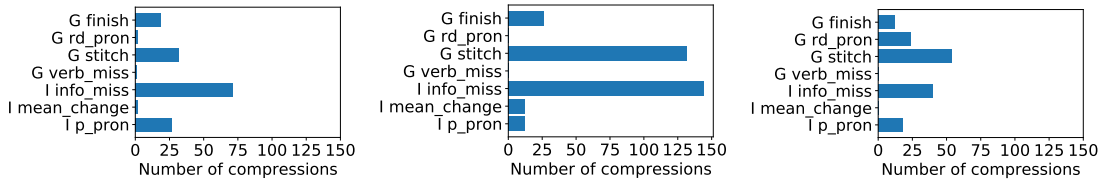
5.4.2 Error analysis

The purpose of error analysis is to find weak spots of a system, from the point of view of human evaluation. In sentence compression, previous work typically analyzed system predictions of the first 200 sentences of the test set, using a 5-point Likert scale to assess annotators' opinions of the compressions' *readability* and *informativeness* (Filippova et al., 2015).

Since error analysis is used for further system improvement and test sets should be used only for final evaluation, we perform error analysis on the development set. In order to do that, we retrieved BERTUNI's predictions on the 200 dev set sentences which received the lowest F1 scores and manually examined them. Note that those are not random samples; the reason why we chose worst predictions is because we know that the system performed poorly on them.

As for the quality criteria, we had to make certain adjustments. Filippova et al. (2015) mention that readability *covers the grammatical correctness, comprehensibility and fluency of the output*, while informativeness measures *the amount of important content preserved in the compression*. We already mentioned that lumping several criteria into one synthetic index is a bad idea, because annotators can't easily decide on the exact facet of evaluation (Section 3.4.1). Given that there already exists a problem of distinguishing fluency and grammaticality, adding both of them to assess readability seems to be a bad design decision. The problem is aggravated by the fact that readability as a text quality criterion is already used by NLP researchers for estimating the *text complexity* from a reader's point of view (Vajjala and Meurers, 2012; Štajner and Saggion, 2013; Venturi et al., 2015; De Clercq and Hoste, 2016). This made us conclude that readability is another overloaded criterion. Instead, we chose *grammaticality* as the first quality criterion.

We manually analyzed BERTUNI predictions on the 200 aforementioned samples, trying to identify common error patterns. The results are presented below.



(a) BERTUNI errors **(b)** BERTBI-TF errors **(c)** Ground-truth errors

Figure 5.5: Number of errors made by the evaluated approaches on the 200 development set instances where BERTUNI achieved the lowest F1 scores, as well as errors found in ground-truth compressions. Error types marked with *G* are *grammaticality* flaws; the remaining ones are errors of *informativeness*.

Grammaticality Out of 200 compressions, 146 (73 %) were deemed to be grammatical. The errors in the remaining instances have been classified into several groups (marked with *G* in Figure 5.5a).

Most of them were cases where grammatical clauses miss linking words and are *stitched* together, making the output ungrammatical, as in the following compressions:

- *I 'm said It 's not Kitty Pryde superhero is the leader of the X-Men .*
- *He first Postal Vote result can be announced before 10PM .*

Another large error category was *finish*: the compression was grammatical until the last retained token, where the sentence ended abruptly, rendering the compression incomplete:

- *Activision Blizzard has confirmed some new statistics for its games including .*
- *The South Sydney star had no case to .*

A few system outputs incorrectly started with a relative or demonstrative pronoun. This happened when the system failed to retain parts of the main clause of a sentence (*rd_pron*):

- *That shows young people rapping while flashing cash and a handgun in a public park .*

Finally, one output missed a verb which was essential for ensuring grammaticality (*verb_miss*):

- *People giant waves crash against the railway line and buildings at Dawlish .*

Informativeness Out of 200 compressions, 105 (52.5 %) were deemed to be informative, the errors in the remaining instances have been classified into several groups (marked with *I* in Figure 5.5a). Most of these erroneous cases were compressions which missed certain information that was needed for understanding the context (*info_miss*). For example:

- **Sentence:** *Dolly Bindra filed a case on an unknown person for having threatened her at gun point today in Oshiwara, Mumbai.*

- **System:** *Dolly Bindra filed a case .*

- **Sentence:** *Mount Hope officially became the third largest city in the state which means jamboree officials are pulling out all the stops.*
- **System:** *Mount Hope became the third largest city .*

A smaller, but still a large group of compressions started with unresolved personal pronouns, which made it hard to understand the subject (*p_pron*):

- **Sentence:** *Britney Spears has said she hopes her latest album Britney Jean will inspire people and she wants to project positive energy out into the world.*
- **System:** *She hopes her album Britney Jean will inspire people .*

- **Sentence:** *A group of lawyers from the Calcutta High Court today came out in support of former Supreme Court judge AK Ganguly, accused of sexually harassing a law intern, saying he should be allowed to work freely till proven guilty.*
- **System:** *He should be allowed to work freely till proven guilty .*

In some cases we noticed that omitting the context caused a change in the meaning of the sentence (*mean_change*):

- **Sentence:** *Serbia's First Deputy Prime Minister Aleksandar Vucic spoke in Germany with former German chancellor Helmut Kohl about Serbia's path towards the EU and its economic recovery/ During the talks, Vucic highlighted the important role of German investors in Serbia and voiced hope that Germany will give even stronger support to Serbia in the realization of its European goals.*
- **Reference:**: *Serbia's Aleksandar Vucic spoke in Germany, Vucic highlighted the important role of investors in Serbia.*
- **System**: *Aleksandar spoke Germany will give stronger support to Serbia .*

A large number of both grammatical and informative compressions did not match references. Interestingly enough, in some cases the system outputs were more grammatical than the references:

- **Sentence:** *In November Newport beat Hartlepool 2-0 at Rodney Parade, we saw their two and raised them to three as we won at the Vic and kept yet another cleansheet in the 3-0 win as the defensive stability continued.*
- **Reference**: *We saw their two and raised to three.*
- **System**: *Newport beat Hartlepool 2 0 .*

5.5 Evaluation Discrepancy

When assessing the sentence compressions, we need to compare system outputs with references. Manual examination revealed that many references themselves are flawed, despite the positive human-based assessment of the dataset quality which we described in Section 5.2. This, in turns, means that noise is inherent to the Google Dataset, and metric-based improvements on this data are misleading. To corroborate this claim, we conducted two experiments: the first one tested the capacity of a more accurate system to ignore the noise and output compressions of better quality. In the second one, we verified whether the noise came from the ground-truth data and attempted to quantify it.

5.5.1 Higher Scores, Worse Quality

At first, we decided to implement more complex models that could potentially achieve better scores. We attempted to improve the grammatical quality of BERTUNI compressions by using the history of model predictions for making more informed decisions. We implemented and tested models that use BERT-encoded lastly-retained tokens at each prediction step as an additional input to the model (prediction history), similar to n-gram language models. As a history, BERTBI and BERTTRI used one and two previously predicted tokens, respectively. BERTBISS and BERTTRISS were the same as BERTBI and BERTTRI, but used scheduled sampling training scheme to mitigate the exposure bias issue (Bengio et al., 2015).

According to the metric evaluation results, none of the more complex models outperformed BERTUNI (Table 5.3).

Model	F1↑	CR↓
BERTUNI	**0.860** ±0.001	0.418 ±0.004
BERTBI	0.849 ±0.001	0.423 ±0.005
BERTBISS	0.840 ±0.003	**0.370** ±0.005
BERTTRI	0.847 ±0.002	0.423 ±0.007
BERTTRISS	0.843 ±0.003	0.382 ±0.006
BERTBI-TF	0.901	0.423

Table 5.3: BERT-based model variants' performance on the development set (mean and standard deviation across ten random seed values). BERTBI-TF was run only once, since it is a "cheating" model that is not meant to be used in production.

We used an unrealistic scenario and artificially made it easier for the model to make correct predictions. We trained a BERTBI-TF model which builds upon BERTBI, but at prediction time for history (i.e. the last two retained tokens) instead of model predictions uses ground-truth labels.[28] The development set result of BERTBI-TF was an F1 score of **0.901**, a 4-point

[28]We call this model BERTBI-TF, since it builds upon BERTBI, but uses teacher forcing (tf) both at training and prediction time.

improvement over BERTUNI. We retrieved this model's predictions for the same 200 dev set sentences used for the error analysis of BERTUNI outputs, and manually examined them.

We assessed BERTBI-TF outputs from the same aspects of *grammaticality* and *informativeness*, as described in Section 5.4.2.

Grammaticality Out of 200 compressions, only 44 (22 %) were found to be grammatical; we classified the errors in the remaining instances into groups (marked with *G* in Figure 5.5b). The first and most prevalent one is the already mentioned *stitch* group which comprises around 80 % of all grammatical errors:

- *The program has received FBS college game 2014 season .*
- *Tskhinvali region with Russia .*

The remaining errors are faulty compression endings (*finish*):

- *The fine has been described as a slap on the .*
- *P Chidambaram sought .*

Informativeness A similar situation was observed when assessing the compressions' informativeness — only 41 (20.5 %) instances were considered as correct. The distribution of errors (marked with *I* in Figure 5.5b) indicates that more than 80 % of cases miss information by omitting important words:

- *Dickinson was a .*
- *Wynalda is mixing .*

A smaller fraction of errors was comprised by the cases with unresolved personal pronouns:

- *He is an education .*
- *It would win 45 to 55 seats in Odisha .*

The remaining errors were the cases where the system compressions changed the semantics of the input:

- **Sentence**: *612 ABC Mornings intern Saskia Edwards hit the streets of Brisbane to find out what frustrates you about other people.*
- **System**: *Saskia frustrates people .*

We counted the cases in which predictions of BERTBI-TF had better or worse quality, compared to BERTUNI. In terms of informativeness, BERTBI-TF improved 15 and worsened 78 instances; in terms of grammaticality, 115 instances were perceived as less grammatical, versus only 13 improved cases, which makes it clear that BERTBI-TF makes many more mistakes than BERTUNI, despite the higher metric scores. We conjectured that this might be caused by the noise in the ground-truth compressions, since similar issues have been reported in the literature in the case of data-to-text generation and machine translation (Nie et al., 2019).

5.5.2 Ground-Truth Noise

In order to verify our findings, we examined the ground-truth compressions in more detail. Only 63 (31.5 %) of these compressions were both grammatical and informative. Figure 5.5c shows a visualization of the error type distribution. Examples of noisy ground-truth compressions are given in Table 5.4.

Error type	Sentence	Ground-truth Compression
finish	Police investigating the unexplained death of a man in Taupo say his van appears to have broken down.	Police investigating the unexplained death say .
	Mortgage fees are going up so where does Pa.	Where does Pa .
rd_pron	POLICE are looking for witnesses after a car was hit by a van which failed to stop on Friday, January 7.	Which failed to stop .
	In a press release, Patrick said Goldstein will be replaced by Rachel Kaprielian, who is currently the state's registrar of motor vehicles.	Who is the state 's registrar .
stitch	Maggie Rose sheds her innocence in her brand new music video for "Looking Back Now."	Maggie Rose sheds for Looking Back Now
	Dolly Bindra filed a case on an unknown person for having threatened her at gun point today in Oshiwara, Mumbai.	Dolly Bindra filed for having threatened her at gun point in Oshiwara Mumbai .
info_miss	Some parents and others in Bessemer City are complaining about a YouTube video that shows [...].	Some parents in Bessemer City are complaining about a video .
	Prime Minister Kevin Rudd has missed the deadline for an August 24 election, with his deputy saying "people should just chill out" about the election date.	People should chill out about the election date .
p_pron	England fast bowler James Anderson does not feel sorry for Australia and has said his team wants to win the Ashes 5-0.	His team wants to win the Ashes 5 0 .
	Armaan will be taken for a medical examination and post that he will be presented in the court today.	He will be presented in the court .

Table 5.4: Manual error analysis results: examples of errors in ground-truth compressions, sampled from 200 development set instances with lowest BERTUNI F1 scores).

The abundant errors related to the use of pronouns in the compressions were predominantly caused by the fact that many instances contained ground-truth compressions with unresolved pronouns; cleaning the data would likely result in better outputs.

The *stitch*, *finish* and *info_miss* errors can be attributed to the fact that many references have missing information or artifacts remaining from the automatic procedure that was used to create these compressions (Filippova and Altun, 2013). Resolving these issues may require more elaborate strategies, beyond simple text substitution.

Conclusion The results of our manual analysis suggest that the compressions are noisy due to the noise in the training data. Strangely enough, this issue has not been mentioned in prior work. Perhaps, the main reason for this is the generally high quality of the system compressions. In this sense, our findings are similar to the observation that BLEU metric scores do not distinguish between high-quality translations (Babych and Hartley, 2008), or that the BLEU metric is miopic to subtle output differences that only human experts can spot (Doddington, 2002). If that is the case, our finding makes it clear that current evaluation protocols need further improvements.

5.6 Chapter Summary

In this chapter we presented a sentence compression case study in which we used the SPF principles to detect an evaluation discrepancy which prior work seems to have overlooked.

In the course of the study, we designed a simple, but effective sequence labeling system based on the Transformer neural network architecture. While the proposed approach achieved the highest scores reported in the research literature, the main message of the experiment is not a higher score — it is the idea that the NLP system evaluation should go beyond comparing metric scores with human judgements.

We found that a higher-scoring system can produce worse-quality outputs. We further provided some empirical evidence that this issue is caused by the noise in the training data. We call this finding a *discrepancy*, because until now no one questioned the quality of the corpus we used in our experiments: the research papers we analyzed described automatic and human evaluation results that overlooked the data quality issue. Of course, it could be that the approaches proposed in previous work produce high-quality sentence compressions, but the absence of error analyses plants a seed of doubt into a reader. In this work, we question not the reported results, but the evaluation approach of previous work. Interestingly, some of the issues are similar to the ones we detected in the E2E NLG Challenge experiments (Chapter 4).

None of the prior works questioned the quality of the data, even though it is known that the dataset was constructed automatically, and therefore should contain noisy examples (as we previously discussed in Section 3.3.1). None of the works ever questioned the use of the compression ratio which seems to be too simplistic to call it a metric that measures the compression effectiveness. As we mentioned before, vague, all-encompassing definitions decrease the quality of human evaluation experiments, making them hard to reproduce, because human assessors focus on various aspects of the criterion (Section 3.5.1).

Finally, the employed sentence compression evaluation protocols do not assume having multiple references. The space of possible compressions in deletion-based sentence compression is bound by sentence length. But because the definition of importance is left out, the candidate space is still very large. The existence of only one reference brings additional requirements for evaluation metrics to work, and commonly used n-gram overlap metrics clearly do not satisfy these requirements.

The presented results showed that the system output analysis is indispensable when assessing the quality of NLP systems. It is a well-established fact that metric scores used for system development do not always reflect the actual quality of a system; usually this is revealed via human evaluation experiments. However, in our case study of automatic sentence compression we have discovered that they might not be sufficient and should be complimented by manual error analysis. Exploratory data analysis of the employed corpora may reveal many of the potential issues in advance and guide the system development process.

This chapter focused on the problem of system evaluation. In the next chapter, we will show how the SPF principles can be used to develop more reliable NLG approaches.

CHAPTER 6

Reliable NLG

In this chapter, we will describe a way to apply the SPF principles to design more reliable NLG approaches. By reliable we mean approaches that *have performance guarantees and transparent decision-making principles*. Interpretability of modern data-driven approaches is one of the biggest issues concerning not only researchers, but also machine learning practitioners who use such techniques in real-world applications. End-to-end neural approaches have achieved state-of-the-art performance in many NLP tasks and have already become a default method of tackling almost any NLG problem. However, they often lack the transparency of the underlying decision-making process, hindering error analysis and model improvements (Samek et al., 2018; Liu et al., 2019a; Carvalho et al., 2019; Danilevsky et al., 2020).

Researchers have proposed a range of techniques to mitigate some of the issues: for example, the attention (Bahdanau et al., 2015), copy (Gu et al., 2016), coverage mechanisms (Tu et al., 2016), as well as architecture modifications, like pointer-generator networks (PGN) (See et al., 2017). These techniques significantly improved the performance of standard seq2seq models. However, the difficulty in the interpretation of model behavior still remains.

In this chapter we argue that one possible way to approach this problem is by *designing a decision-making algorithm* and letting a neural model to only make intermediate decisions: the algorithm defines possible actions which change the state of the system; the task of the neural model is to make a decision as to which action to take, based on the current state and the history of previous decisions.

Using surface realization as an example task, we analyze its specifications and, based on the relation between the task and data, develop a system that not only achieves state-of-the-art performance, but also exhibits more interpretable behavior.

6.1 Multilingual Surface Realization: Overview

For many years, one of the research foci in the NLG community has been surface realization (SR), *the process of transforming a sentence plan into a linearly-ordered, grammatical string of morphologically inflected words* (Langkilde-Geary, 2002). However, until recently, the differences

in the input representations and levels of input preprocessing complicated the comparison of surface realization systems.

The first collaborative effort in unifying the task specifications started almost a decade ago, with the first shared task on surface realization (Belz et al., 2011). It took place in 2011 and was aimed at *developing a common input representation* that could be used by a variety of NLG systems to generate realizations from. Following the division between more semantic-oriented and text-oriented NLG systems, the task organizers developed two NLG input types, created by post-processing the CoNLL 2008 Shared Task data (Surdeanu et al., 2008): *shallow* syntactic dependencies and *deep* representations which were built on top of the shallow ones, but enriched with the semantic annotations from Nombank (Meyers et al., 2004) and Propbank (Palmer et al., 2005). The main result of the SR'11 task was standardization of the surface realization inputs at two levels, facilitating further development of off-the-shelf NLG tools.

In 2018, the first multilingual version of the challenge was organized (Mille et al., 2018). This time, the organizers took advantage of the availability of multilingual treebanks annotated with Universal Dependencies (UD) (De Marneffe et al., 2014). Following the technological advances in the machine learning community, SR'18 also adapted its agenda: the task was organized in order to encourage the perspective participants *to explore the extent to which neural network parsing algorithms can be reversed for generation*. The task also addressed questions about the suitability and usefulness of UD for NLG.

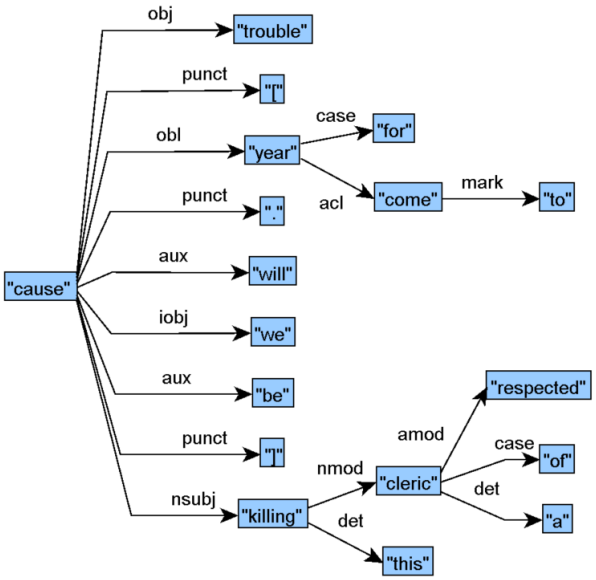
The latest edition of the shared task took part in 2019 (Mille et al., 2019). In order to improve the comparability of systems, the organizers focused on *the closed task setting* and did not allow other annotated data than the one provided by the organizers to be used for training the systems.

We took part in the SR'18 task and therefore focus on its specifications. Following the SR'11 protocol, the organizers proposed to use two representations, depending on the track the contestants participated in.

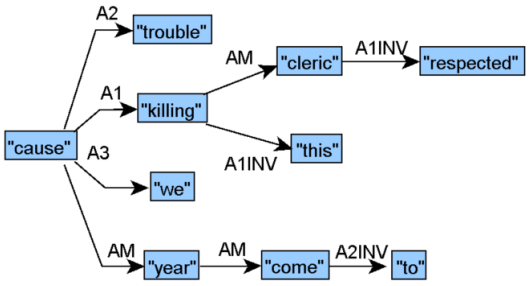
Shallow Track The participants started with unordered dependency trees consisting of lemmatized nodes with POS tags and morphological information as found in UD annotations (version 2.0). A graphical representation of a sample input and the corresponding output sentence are shown in Figure 6.1a. The task in this track was to determine the correct order of lemmas and inflect them to generate surface forms. The data was available for the following ten languages: Arabic, Czech, Dutch, English, Finnish, French, Italian, Portuguese, Russian and Spanish.

Deep Track The inputs were similar to the Shallow Track, but had a higher level of abstraction. For example, functional words (auxiliaries, functional prepositions and conjunctions) and surface-oriented morphological and syntactic information were removed (Figure 6.1b). In addition to what was required for the Shallow Track, the task in the Deep Track thus was also to reintroduce the removed functional words and morphological features. However, the representation was available only for three languages: English, French and Spanish.

The researchers that advocate for using more abstract input representations claim that such inputs are closer to a realistic application context for modern NLG systems, where the text gener-



(a) Shallow Track



(b) Deep Track

Figure 6.1: The SR'18 Shallow and Deep Track representations of a sample UD sentence: *[This killing of a respected cleric will be causing us trouble for years to come.]*.

ation component does not have access to syntactic or language-specific information (Mille et al., 2018). In this sense, the SR'18 Deep Track follows a range of other data-to-text generation tasks that took place recently: the already mentioned E2E NLG Challenge, as well as WebNLG (Gardent et al., 2017) and the SemEval-2017 Task on Abstract Meaning Representation Parsing and Generation (May and Priyadarshi, 2017).

However, such meaning representations tend to be less consistent and prone to semantic inadequacy issues, since much of the content is left underspecified (Section 2.2.2). For example, the SR'18 Deep Track input words are not sense-disambiguated, full prepositions and some argument relations may be missing. This could lead to potential issues with system outputs, like hallucination or content dropping (Section 3.3.1). For this reason, we focused on the Shallow Track, and, therefore, our goal was to design an approach to generate sentences by ordering the lemmas and inflecting them to the correct surface forms.

6.2 Data Analysis

Prior to system development, we analyzed the data along the dimensions which we considered relevant for the task. As an illustration, we show figures and numbers for English, since that was the primary language we used in the development process. The analysis results for the other languages can be found in (Puzikov and Gurevych, 2018a).

Data size The shared task can be considered as operating under a low-resource scenario: Table 6.1 shows that SR'18 treebanks are rather small. This means that using popular neural seq2seq approaches would be challenging, since they require large amounts of training data to learn useful data representations (Koehn and Knowles, 2017; Hou et al., 2018; Simpson and Gurevych, 2019).

	Language									
	ar	cs	en	es	fi	fr	it	nl	pt	ru
Train	6 016	66 485	12 375	14 289	12 030	14 529	12 796	12 318	8 325	48 119
Dev	897	9 016	1 978	1 651	1 336	1 473	562	720	559	6 441
Test	676	9 876	2 061	1 719	1 525	416	480	685	476	6 366

Table 6.1: Number of training, development and test sentences in SR'18 datasets.

References For the input to the Shallow Track, the organizers separated the reference sentences from the respective structures. Although the one-to-one correspondence between sentences and dependency trees was preserved, the alignment between the lemmas in the trees and the word forms in the sentences was lost. To circumvent this issue and ease the burden of aligning lemmas with the corresponding surface forms, we decided to use the original UD data files for all our experiments — they contain the same dependency trees as the shared task data, but the order of the tokens is not scrambled and each surface form is aligned with the respective lemma.

Lemma-form statistics We examined the number of possible forms per lemma (Figure 6.2). The majority of lemmas have only one surface form, which suggests a strong majority baseline for the morphological inflection subtask. However, languages with rich morphology (Czech, Finnish, Russian) pose a challenge in this regard and call for a more elaborate approach which takes into account complex grammar inflection paradigms. The number of unique features (values in the *FEAT* column of the input data) served as a rough estimate of the latter (Table 6.2).

OOV The number of out-of-vocabulary (OOV) language units can be viewed as a crude measure of the expected difference between training and development data distributions. Table 6.2 shows the number of OOV lemmas, surface forms and characters for each of the languages. As expected, a large number of morphological variants in Russian and Czech result in much larger vocabulary sizes and OOV counts. Some of the datasets included foreign names and terms which are used in their original language forms. For example, out of 356 464 French data tokens, 419 include

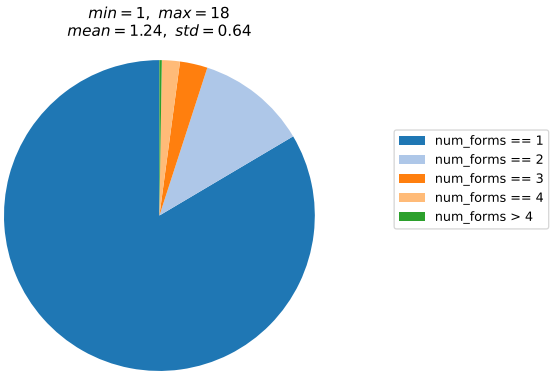


Figure 6.2: The number of possible forms per lemma (English SR'18 training set).

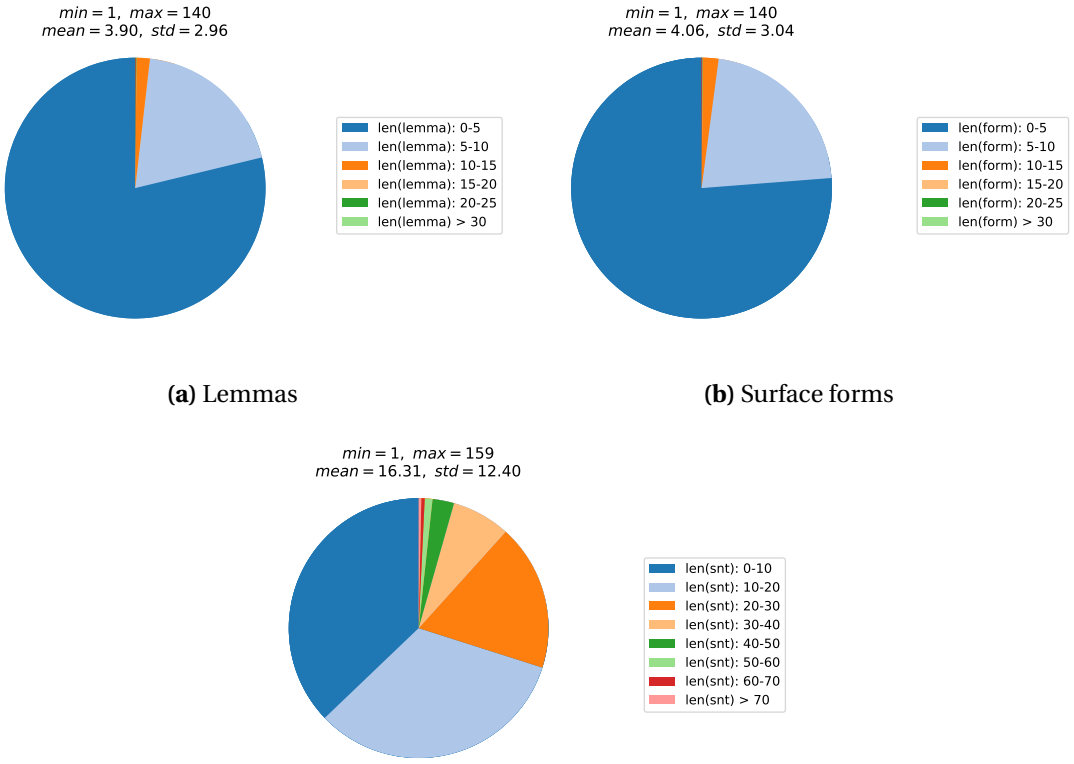
	Language									
	ar	cs	en	es	fi	fr	it	nl	pt	ru
Unique features	37	**112**	36	56	89	35	41	66	48	40
OOV lemmas	1 056	**3 299**	1 180	1 368	1 598	1 895	439	973	535	2 723
OOV forms	1 745	8 070	1 313	2 131	3 666	2 387	683	1 131	785	**8 190**
OOV chars	0	2	3	1	5	**12**	2	0	0	0

Table 6.2: Cross-lingual data analysis (SR'18 training set). Out-of-vocabulary (OOV) statistics computed over the development set, with respect to the training data. Bold-faced numbers correspond to the most challenging language cases.

characters that are not digits, punctuation signs or letters of the French alphabet. Since such words are usually not conjugated, but copied verbatim, we consider them as outliers and exclude them from the training procedure. In our initial experiments, tokens defined in the UD annotation guidelines as multi-word expressions (MWE) and empty nodes were excluded from the training data, because they require language-specific treatment (e.g. the French data includes 9 750 tokens which were identified as MWE; out of 870 033 tokens in the Russian dataset, 1 092 correspond to empty nodes). However, in a subsequent pilot study, we show that a proper treatment of these language phenomena increases the performance of the system (Section 6.5.4).

Length distributions Another important property of the data is the length distribution of lemmas, surface forms and sentences (Figure 6.3). We computed the training data statistics and used the obtained estimates to establish cut-off thresholds for filtering out outlier lemmas and forms from the training data.

Syntactic complexity When approaching the task of syntactic ordering, one needs to take into account the complexity of the tree structures. We found the branching factor to be very informative



(a) Lemmas

(b) Surface forms

(c) Sentences

Figure 6.3: Length distributions of lemmas, surface forms and sentences (SR'18 English training set).

in this regard: for each node in each tree, we counted the number of children the node has.[29] Most nodes in the dependency trees of all examined languages have from one to three children (Figure 6.4). This solicits decomposition of the syntactic ordering procedure over subtrees, similar to what was done in (He et al., 2009).

Task-data checklist Below is the task–data relation checklist which we filled based on our observations.

Task and Data

- Data annotation:

[29] In this thesis we will interchangeably use the terms *dependent* and *child*, when mentioning the end nodes of a dependency relation. Similarly, *parent*, or *head*, nodes are those that govern the relation.

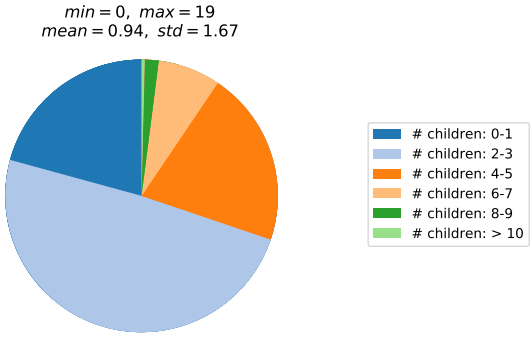


Figure 6.4: Branching factor of the dependency trees (English SR'18 training set).

- ☑ Is the phenomenon at hand ambiguous? **Yes: generation from a dependency tree may result in several valid output sentences (Filippova and Strube, 2009).**
- ☑ Are the annotation guidelines simple? **No.**
- ☑ Are the guidelines ambiguous? **No.**
- ☑ Do annotators have the needed expertise? **Yes.**
- ☑ How many annotators are there per data point? **Varies: UD is a community effort.**
- ☑ Does the data contain time-dependent phenomena? **Varies: depends on the treebank.**

- Data artifacts:
  - ☑ Is the data noisy? **No.**
  - ☑ Could the data be biased? **No.**
  - ☑ Is it a low-resource scenario? **Yes.**

Note that surface linearization from one lemmatized dependency tree may result in several valid output candidates (Filippova and Strube, 2009). This situation is more common for languages with flexible word order. An illustration of this for Russian is shown in Figure 6.5: three different sentences with the same meaning can be generated from one lemmatized dependency tree.

Conclusion UD is an open community effort with over 300 contributors producing nearly 200 treebanks in over 100 languages. While the annotations guidelines[30] are rather complex and the

[30]https://universaldependencies.org/guidelines.html

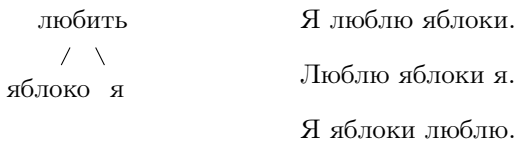


Figure 6.5: An illustration of output variation in surface realization (Russian): a simple lemmatized dependency tree (left) and three semantically equivalent sentences which can be generated from this tree.

annotated phenomena can be ambiguous, we do not expect the treebanks to contain many errors or inconsistencies, because the core members that develop the guidelines are NLP experts motivated to create high-quality corpora, and the proposed annotations are actively moderated by the UD administration.

From the data analysis perspective, the biggest challenge is presented by small treebank sizes, large space of surface forms in the case of morphologically rich languages, and a large number of OOV tokens (Table 6.2).

6.3 Related Work

The general approach to the Shallow Track task is to generate a sentence by ordering the lemmas and inflecting them to the correct surface forms. Past research work proposed both joint and pipeline solutions for the problem.

Syntactic Oordering Given a bag of input words, a syntactic ordering algorithm constructs an output sentence. Prior work explored a range of approaches to syntactic ordering: grammar-based methods (Elhadad and Robin, 1992; Carroll et al., 1999; White et al., 2007), generate-and-rerank approaches (Bangalore and Rambow, 2000; Langkilde-Geary, 2002), tree linearization using probabilistic language models (Guo et al., 2008).

Conceptually, the problem of tree linearization is simple. However, given no constraints, the *search space is exponential in the number of tokens*, which makes exhaustive search intractable (Zhang, 2013). This stimulated the line of research focusing on the development of approximate search methods. Current state-of-the-art (evaluated on the Penn Treebank only) belongs to the system of Puduppully et al. (2016) who extended the work of Liu et al. (2015) on developing a transition-based generator. The authors treated language generation process as a generalized form of dependency parsing with unordered token sequences, and used a learning and search framework of Zhang and Clark (2011) to keep the decoding process tractable. A similar approach to dependency tree linearization was explored in (Bohnet et al., 2010), who approximated exact decoding with a beam search.

Our proposed method of syntactic ordering is also based on search approximation, but follows a different approach: we use a greedy search strategy, but restrict the scoring procedure to a smaller

set of plausible candidate pairs, which speeds up the search process and reduces the number of mistakes the system might make.

Morphological inflection Word inflection in the context of the Surface Realization Task can be defined as the subtask of generating a surface form (*was*) from a given source lemma (*be*) and additional morphological or syntactic attributes (*Number=Sing*, *Person=3*, *Tense=Past*).

Early work proposed to approach the task with finite state transducers (Koskenniemi, 1983; Kaplan and Kay, 1994). While being accurate, these systems require a lot of time and linguistic expertise to construct and maintain. With the advance of machine learning, the community mostly shifted towards data-driven methods of automatic morphological paradigm induction and string transduction as the method of morphological inflection generation (Yarowsky and Wicentowski, 2000; Wicentowski, 2004; Dreyer and Eisner, 2011; Durrett and DeNero, 2013; Ahlberg et al., 2015). In comparison with their rule-based counterparts, these approaches scale better across languages and domains, but require manually-defined comprehensive feature representation of the inputs.

Current research focuses on data-driven models which learn a high-dimensional feature representation of the input data during the optimization procedure in an end-to-end fashion. Recent work (Faruqui et al., 2016) proposed to model the problem as a seq2seq learning task, using the encoder-decoder neural network architecture developed in the machine translation community (Cho et al., 2014c; Sutskever et al., 2014). This approach showed an improvement over conventional machine learning models, but failed to address the issue of poor sample complexity of complex neural networks — in practice, the approach did not perform well on low-resource or morphologically rich languages.

An attempt to address this issue was made by Aharoni and Goldberg (2017), who proposed to directly model an almost monotonic alignment between the input and output character sequences by using a controllable hard attention mechanism which allows the network to jointly align and transduce, while maintaining a focused representation at each step. The authors proposed to utilize independently learned character-level alignments instead of the weighted sum of representations (as done in the soft attention models). Experimental results demonstrated better sample efficiency of the models trained according to the proposed method, and considerable improvements over the previous approaches.

6.4 Binary Linearization

Motivation The results of our data analysis (Section 6.2) revealed several challenges. First of all, small corpora sizes essentially prohibit us from using seq2seq neural networks. One way to alleviate this issue is to *lift the burden of making strategic decisions from the data-driven algorithm by outlining a linearization algorithm*, whereby the model only needs to make tactic decisions. This approach follows the already mentioned tradition of hybrid NLG systems (Section 2.3.1). Similar approaches have been developed in other NLP areas as well. For example, transition-based dependency parsing algorithms (Nivre, 2003; Yamada and Matsumoto, 2003; Attardi, 2006) rely

on a locally trained classifier for predicting the next parser action given a compact representation of the derivation history. The possible actions, however, are provided by a human-designed transition system. Another example is the copy mechanism by which a data-driven system chooses to generate something from the probability distribution of the training data, or copy the currently processed unit (See et al., 2017). The efficiency of such approaches has been linked to the decomposition of the decision-making process (Sartorio et al., 2013).

We decided to explore this direction as well, and develop a modular system that has a human-designed algorithm for making strategic decisions during surface realization. Apart from being more data-efficient, it has another benefit: it separates the decisions made by a data-driven system from the algorithm transitions. Compared to seq2seq approaches which are still considered to be black-box, hybrid systems of the kind we mentioned above exhibit more understandable and reliable behavior (Carvalho et al., 2019).

At a high-level, we propose to approach the task in two steps:

(1) **Syntactic ordering**: use a syntactic ordering algorithm and a data-driven classifier to convert a dependency tree into a binary tree, then traverse the latter to obtain a sequence of lemmas.
(2) **Morphological inflection**: conjugate each lemma into a surface form.

During the first stage, the system separates the input dependency tree into subtrees of depth one and incrementally converts each of them into an equivalent binary subtree. Once all subtrees have a binary tree equivalent, a deterministic procedure merges them into the final binary tree. Since each node is labeled with the corresponding lemma, we simply run an in-order traversal over the resultant binary tree to obtain a lemma sequence.

In the second step, we use a trained morphological inflection generator component to predict a surface form for each lemma in the sequence; by doing so, we obtain an output sentence. We chose the aforementioned approach of Aharoni and Goldberg (2017) as our morphological inflection method.

6.4.1 Syntactic Component

The first step of the proposed pipeline orders the nodes of the dependency tree into a sequence which ideally mirrors the order of words in the reference sentence. The main difficulty of this step is finding a sorting or ranking method which avoids making many node comparisons or scoring decisions. We propose an ordering procedure which uses a given dependency tree and constructs a binary tree storing the original dependency nodes (lemmas) in a sorted order. We call this procedure *binary tree–based linearization*, or BINLIN (Algorithm 1).

Linearization algorithm As input, the algorithm takes a dependency tree and a classifier trained to make binary decisions of positioning child nodes to the right or left of the head node. First, we decompose the tree into local subtrees, represented by *(head, children)* node groups. This is achieved by running a breadth-first search (BFS) algorithm on the input dependency tree

Algorithm 1 Binary Linearization

Input: dependency tree dt, trained binary classifier clf
Output: ordered sequence of lemmas $oseq$

```
1:  function BinLin(clf, dt)
2:      subtrees ← dict
3:      nodes_bfs ← BFS(dt)                          ▷ breadth-first search
4:      for head, children ∈ reverse(nodes_bfs) do   ▷ process nodes bottom-up
5:          bt ← InitBinTree(head)                   ▷ initialize binary subtree
6:          for ch ∈ children do
7:              InsertNode(bt, ch, clf)              ▷ insert child nodes
8:          subtrees[head] = bt                      ▷ store the binary subtree
9:      bt_final ← Merge(subtrees)                   ▷ merge binary subtrees
10:     oseq ← IoTraverse(bt_final)                  ▷ binary tree in-order traversal
11:     return oseq
```

(line 3 of the pseudocode). Since we build the binary tree bottom-up, we reverse the order of the node-groups, and start processing node groups from the leaf nodes.

For each *(head, children)* group, we further apply the following steps:

(1) Initialize a binary tree with the head node (line 5).
(2) Iterate over the child nodes and use the classifier to predict whether the child should be inserted to the left or to the right of the head node (lines 6-7).

Once all the children have been inserted, the construction of a binary subtree is finished and the algorithm moves on to the next dependency subtree. We process the nodes in a bottom-up manner, building binary subtrees until we reach the root node. At this point, we have processed all the nodes from the original dependency tree, and the binary subtrees can now be merged to construct the final binary tree with the root as a head node. Figure 6.6 shows an example dependency tree and an equivalent binary tree that we construct when applying Algorithm 1. When the binary tree construction is finished, we can obtain a sorted lemma sequence by performing in-order traversal on the resulting binary tree.

The decision-making core of the procedure is the insertion of a new node into the binary tree (Algorithm 2).

Classifier Given a pair of nodes (n_i, n_j), we first extract their features. Our feature set encodes both local and global contexts. The former is a representation of the node pair under consideration, while the latter considers all the nodes in the corresponding subtree.

When computing a node feature representation vector, we use the following feature sources: the node itself, it's head and one (any) child in the dependency tree as the neighborhood elements. We extract the corresponding lemmas, POS tags[31], the dependency edge labels, the number of

[31] *XPOS* and *UPOS* columns in the CoNLL input data file.

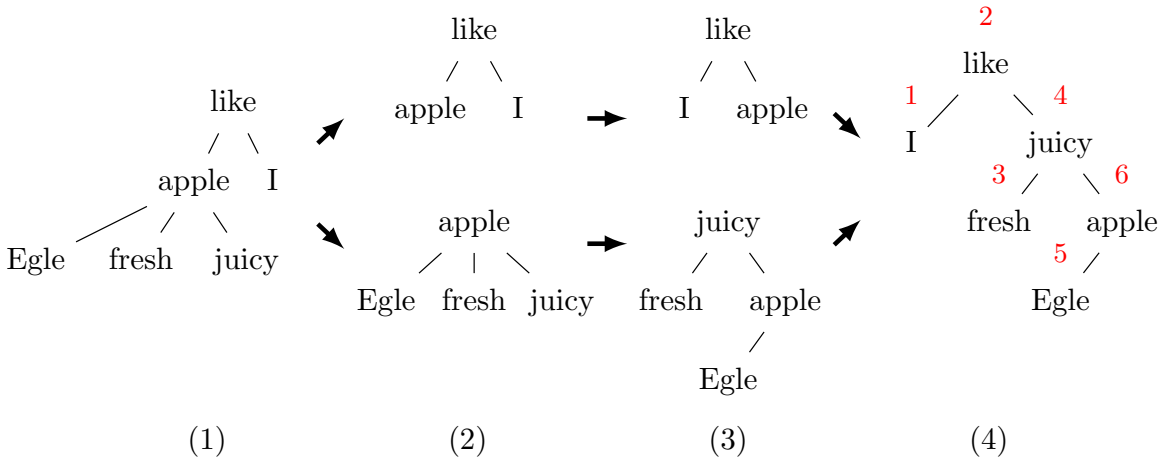


Figure 6.6: A high-level overview of the BINLIN algorithm. Decompose the dependency tree (1) into subtrees of depth one (2), then convert subtrees into equivalent binary trees (3), and merge them. In-order traversal of the merged binary tree (4) produces a sequence of lemmas *I like fresh juicy Egle apple*.

the node's children, and the length of the path from the node to the root of the dependency tree. Thus, the feature set F for one node in the node pair consists of $N = 3$ (neighborhood elements) × 6 (features) = 18 components.

Each feature is represented as a d-dimensional embedding vector, each node n_k is represented as a vector $\mathbf{x}_k \in \mathbb{R}^{N \times d}$, where N denotes the number of extracted features and d is the embedding size. In other words, each dense node representation $\mathbf{x}_k$ is a concatenation of the embeddings for each feature in the feature set F. The embedding matrix is denoted as $\mathbf{E} \in \mathbb{R}^{d \times |V|}$, where V is the vocabulary of unique lemmas, POS tags and dependency edge labels, observed in the training data.

For simplicity, we decided to use a multi-layer perceptron as a binary classifier (Figure 6.7).

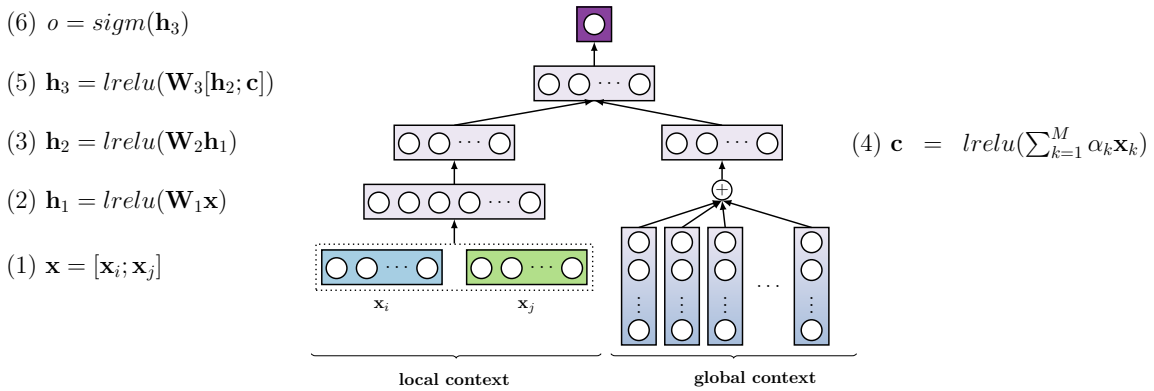


Figure 6.7: Schematic view of the neural network architecture used as a classifier for the syntactic ordering component of BINLIN.

Algorithm 2 Recursive Node Insertion

Input: binary tree bt, new node $node$, trained binary classifier clf

```
1: procedure INSERTNODE(bt, node, clf)
2:     bf ← GetFeat(bt)                          ▷ extract head node features
3:     cf ← GetFeat(child)                       ▷ extract child node features
4:     label ← MakeDecision(clf, cf, bf)         ▷ classify and make decision
5:     if label is LEFT then                     ▷ insert node
6:         if bt.left is None then
7:             bt.left ← BinTree(child)
8:         else
9:             InsertNode(bt.left, child, clf)
10:    else
11:        if bt.right is None then
12:            bt.right ← BinTree(child)
13:        else
14:            InsertNode(bt.right, child, clf)
15: function MAKEDECISION(clf, cf, bf)
16:    score ← clf(cf, bf)
17:    if score ≥ clf.threshold then
18:        decision ← RIGHT
19:    else
20:        decision ← LEFT
21:    return decision
```

The embedding vectors for the two nodes under consideration ($\mathbf{x}_i$ and $\mathbf{x}_j$) are further concatenated to form the input to the classifier (Figure 6.7, step 1), and projected onto a lower-dimensional space via two non-linear transformations (Figure 6.7, steps 2 and 3). In all our experiments, we use Leaky ReLu as a non-linear function (Maas et al., 2013)

The global context is computed as a weighted sum of feature representations of the nodes in the subtree (Figure 6.7, step 4). The computation is done similar to the attention mechanism of Bahdanau et al. (2015).

The local and global contexts are combined via another non-linear transformation (Figure 6.7, step 5), and form input to the last layer of the network with a logistic regression function on top to predict the probability of node n_j being positioned to the right ($y = 1$) or left ($y = 0$) of node n_i in a binary tree:

$$p(y = 1) = sigm(\mathbf{h}_3; \theta) = \frac{1}{1 + e^{\theta \cdot \mathbf{h}_3}} \tag{6.1}$$

θ denote the parameters of the classifier. The decision-making rule is defined by setting a threshold on the output of $p(\cdot)$:

$$decision = \begin{cases} \text{right}, & \text{if } p(y=1) \geq 0.5, \\ \text{left}, & \text{otherwise.} \end{cases} \tag{6.2}$$

Training data preparation In order to prepare training data for the classifier, we consider each (n_i, n_j) node pair extracted from subtrees of depth one. For each $(n_i, n_j, label)$ triple originally present in a tree, we add (n_j, n_i, op_label) with node positions flipped and having the opposite label. In our running example *I like Egle apple*, one of the training data points we have is (*like*, *I*, *LEFT*), since the word *I* in the sentence is positioned to the left of its head *like*. In addition, we add the (*I*, *like*, *RIGHT*) triple to the training set.

We motivate this procedure by the fact that there is no guarantee that our system learns position-invariant word order representations, i.e. if node n_j should be positioned to the right of n_i, then n_i should be inserted to the left of n_j. It is known that neural networks do not have this reasoning ability (Shorten and Khoshgoftaar, 2019; Hu et al., 2019; Wei and Zou, 2019), and in our preliminary experiments we verified that this is indeed the case. In order to circumvent this issue, we explicitly add symmetric node pairs with opposite labels to the training set.

The syntactic ordering component was implemented using feed-forward networks; the corresponding hidden layer sizes were fixed as follows: at 100, 64 and 264 dimensions (weight matrices $\mathbf{W}_1$, $\mathbf{W}_2$ and $\mathbf{W}_3$ in Figure 6.7). We used 200-dimensional randomly initialized word-level case-sensitive embeddings.

6.4.2 Morphological Component

To create a sentence from an ordered sequence of lemmas, we need to predict the correct morphological form for each of them. This is the purpose of the second component of our system. While we focused mostly on the syntactic realization component, as part of the system development we experimented with the following three different morphological inflection models:

- MORHPMLP: a simple multi-layer perceptron similar to the one employed for the syntactic component;
- MORHRNNSOFT: an encoder-decoder architecture with an attention mechanism of Bahdanau et al. (2015);
- MORPHRNNHARD: an encoder-decoder model with a hard monotonic attention of Aharoni and Goldberg (2017).

The morphological inflection component was implemented using character-level LSTM networks, a two-layer bidirectional encoder and a one-layer unidirectional decoder with hidden layer sizes of 200. We used 200-dimensional randomly-initialized embeddings. We did not lowercase the data, maintained a fixed size vocabulary of 250 characters, and for training used only sequences of maximum 30 characters long (both source and target sides).

OOV lemmas and characters during decoding were copied without any changes.

6.5 Experiments

All results are computed on tokenized data instances. Training data was filtered to exclude outliers according to the results of the data analysis (Section 6.2). All neural network components were implemented using PyTorch. The syntactic component used 200-dimensional randomly initialized token-level embeddings, 100-dimensional projection and 64-dimensional hidden layers. The morphological component used 100-dimensional randomly initialized character-level embeddings, a 2-layer bi-directional encoder and 1-layer uni-directional decoder, both using 100-dimensional LSTM cells.

The syntactic and morphological components were trained separately using the Adam optimizer with a learning rate of 0.001, minimizing cross-entropy loss per predicted token. We trained the system with a batch size of 600 for the syntactic component and 200 for the morphological component, for a fixed number of epochs (10 for the syntactic component and 15 for the morphological one).[32] Model selection was done based on the loss value computed on the development set.

The task used two quality criteria, *readability* and *meaning similarity*. The former criterion was rather vaguely defined as "sometimes called as fluency, [it assesses] how well the given text reads; is it good fluent English, or does it have grammatical errors, awkward constructions, etc.". The second criterion measured how close in meaning the outputs are to the original sentence (Mille et al., 2018).

For automatic evaluation, the organizers used smoothed BLEU-4, NIST, and inverse normalized character-based string-edit distance (in our thesis we refer to it as i-EDIST).[33] Surface realization is a sentence-level task, which is why it is not clear why metrics like TER were not included in the evaluation setup. It is strange, because SR'11 included both METEOR and TER, which have been shown to correlate most with human judgments of various aspects of textual quality (Section 2.4.2).

Intrinsic metric-based evaluation has been shown to produce reliable results in the task of surface realization, since corpus-based techniques are well suited to evaluate grammatical coverage (Reiter and Belz, 2009). The authors also note that the acceptable output candidates do not exhibit much variation, which means that a few reference texts most likely adequately cover the solution space.

Metric-related checklists Taking this into consideration, we fill in the metric-related checklists as outlined below.

Task and Evaluation Criteria

[32] Batch size and number of epochs were different due to the constraints of the computational budget.

[33] The organizers chose to use the name DIST instead. We found this to be misleading, since the metric is inverse of a distance.

- Task-Criteria Match:
  - ☑ Do the criteria reflect the specifics of the task? **Yes.**
- Criteria scope:
  - ☑ Is each individual criterion focused on a single quality dimension? **No, readability seems to have been merged with fluency and grammaticality.**

Evaluation Criteria and Metrics

- Targeted Language Level:
  - ☑ Does the metric measure the correct linguistic level? **Yes.**
- Task Specificity and Metrics:
  - ☑ Is the metric supposed to approximate the chosen evaluation criterion? **Yes: see (Reiter and Belz, 2009).**

Data and Metrics

- Metric Assumptions:
  - ☑ Does the metric assume low quality of the outputs? **BLEU – yes, not enough evidence on the rest of the metrics.**
  - ☑ Is the metric supposed to work on corpus/segment level? **Corpus.**
  - ☑ Has the metric migrated from another task? **Yes.**
- References:
  - ☑ Is the metric sensitive to the reference quality? **Yes.**
  - ☑ Is the number of references sufficient? **Varies: some languages permit several valid output candidates.**

Metric Considerations

- ☑ Are the compared approaches typologically different? **No: see Section 6.5.3.1.**
- ☑ Is there any evidence that the metric correlates well with human judgements of the measured quality in the task at hand? **Yes: see (Reiter and Belz, 2009)**.

6.5.1 Morphological component

We start with the evaluation of the morphological inflection generator. It was trained separately ten times with different random seeds. We report the exact string match accuracy for each of the tested approaches. Table 6.3 shows mean scores and standard deviation for each model evaluated on the development data and averaged across the random seed values.

6.5.1.1 Results

	Language				
	ar	cs	en	es	fi
Accuracy (uncased)					
LEMMA	13.47	56.43	85.47	71.45	44.43
MAJOR	69.15	63.50	86.80	76.13	51.04
MORPHMLP	86.63 ± 0.507	94.40 ± 0.052	96.41 ± 0.053	96.72 ± 0.151	78.26 ± 0.217
MORPHRNNSOFT	88.48 ± 2.409	96.61 ± 0.598	93.57 ± 1.370	97.20 ± 0.801	81.05 ± 7.405
MORPHRNNHARD	**93.07** ± 0.515	**99.53** ± 0.031	**98.11** ± 0.054	**99.59** ± 0.027	**95.46** ± 0.923
Accuracy (cased)					
MORPHMLP	86.63 ± 0.507	87.31 ± 0.083	88.79 ± 0.169	93.52 ± 0.195	71.90 ± 0.286
MORPHRNNSOFT	88.48 ± 2.409	89.98 ± 0.638	86.32 ± 1.446	94.15 ± 0.823	75.65 ± 6.869
MORPHRNNHARD	**93.07** ± 0.515	**93.07** ± 0.047	**90.76** ± 0.186	**96.60** ± 0.037	**89.32** ± 0.861

(a) Languages: Arabic, Czech, English, Spanish, Finnish

	Language				
	fr	it	nl	pt	ru
Accuracy (uncased)					
LEMMA	70.44	67.88	79.35	74.19	50.06
MAJOR	74.02	72.48	82.74	75.85	55.64
MORPHMLP	92.73 ± 0.094	94.09 ± 0.062	91.05 ± 0.110	94.12 ± 0.198	90.43 ± 0.122
MORPHRNNSOFT	92.30 ± 0.797	92.54 ± 3.721	85.82 ± 1.993	94.27 ± 3.424	93.65 ± 2.980
MORPHRNNHARD	**95.56** ± 0.066	**97.44** ± 0.240	**95.68** ± 0.115	**99.30** ± 0.035	**98.22** ± 0.056
Accuracy (cased)					
MORPHMLP	88.17 ± 0.128	89.54 ± 0.085	85.79 ± 0.236	89.90 ± 0.171	83.32 ± 0.152
MORPHRNNSOFT	87.90 ± 0.803	88.05 ± 3.524	80.69 ± 1.903	90.06 ± 3.379	87.70 ± 2.763
MORPHRNNHARD	**91.24** ± 0.099	**93.08** ± 0.296	**90.58** ± 0.219	**95.19** ± 0.119	**92.32** ± 0.079

(b) Languages: French, Italian, Dutch, Portuguese, Russian

Table 6.3: Evaluation of the morphological inflection system component on the original UD development set using the percentage of exact string matches as a metric. For the neural architectures, we report both case-sensitive and case-insensitive mean scores and standard deviation (averaged across ten random seed values).

Two simple baselines were developed for the experiment: given a lemma, LEMMA copies the lemma itself as a prediction of the surface form, MAJOR outputs the most frequent surface form if the lemma is not an OOV item, or the lemma itself, otherwise. Lemma-form frequencies were

computed on the training data. For the baselines, we report case-insensitive scores only: their purpose was to sanity-check the development of data-driven morphological components.

As expected, the baselines are outperformed by all data-driven methods examined. Strong performance of the majority baseline for English and Dutch data can be attributed to the simpler morphology of the languages.

The best results are achieved by MORPHRNNHARD, which outperforms all other methods across all languages. Despite the fact that the approach has a bias towards languages with concatenative morphology (due to the assumption of the monotonic alignment between the input and output character sequences), it also performs well on Arabic. This model was chosen for our further pipeline experiments.

Bad sample complexity of MORPHRNNSOFT, the soft attention model, explains its inferior performance compared to the hard attention model. MORPHRNNSOFT model seems to be highly sensitive to the different values of hyper-parameters; its performance has the highest standard deviation among all models, which is most likely due to the same sample complexity issue. Interestingly enough, on English, French, Italian and Dutch data the multi-layer perceptron architecture (MORPHMLP) achieves better results. The latter has a considerably simpler, but less flexible structure, which prohibits the usage of such networks for languages with rich morphology — the number of parameters needed to account for various forms and morphological features grows rapidly until the model can no longer fit into the memory. This also highlights the importance of cross-lingual evaluation of morphological analyzers and generators.

6.5.1.2 Error analysis

In order to better understand the most common errors made by each of the approaches (excluding the baselines), we examined the predictions of the models on the English development set. We filtered out incorrect predictions of capitalization of the first letter of the word, because these cases are ignored by the official evaluation protocol. After the filtering, we randomly sampled 100 erroneous predictions and manually examined them; the results are shown in Table 6.4.

Error types	MORPHMLP	MORPHRNNSOFT	MORPHRNNHARD
wrong lemma	42	-	-
wrong form	29	8	26
alt. form	29	17	57
non-exist. form	-	29	4
proper noun err.	-	27	-
wrong digit seq.	-	13	-

Table 6.4: Major error types made by each of the tested morphological component models.

Unlike character-based models, MORPHMLP treats each surface form as an atomic unit and is therefore prone to errors caused by the data sparsity issues, failing to predict correct forms for unseen lemmas or unseen grammar patterns (*wrong lemma* error type). If the model correctly

identifies the base form and still makes a mistake, in half of the cases it is an incorrect prediction of verb tenses, singular/plural noun forms or indefinite English articles (*wrong form*). The latter cases are caused by the fact that our model does not use any information about the next token when predicting the form of the current lemma. This limitation is inherent to the pipeline architecture we employed and can be accounted for in a joint morphology and syntax modeling scenario. Finally, there are also cases where a model predicts an alternative surface form which does not match the ground truth, but is grammatically correct (*alt. form*): *not*/*n't*, *are*/*'re*, *have*/*'ve*. Strictly speaking, the latter cases are not errors, but for simplicity we will treat them as such in this section.

MorphRnnSoft model predicts fewer wrong morphological variants, but suffers from another problem — hallucinating non-existing surface forms: *singed*, *dened*, *siaseme* instead of *sung*, *denied*, *siamese*. This is not surprising, given the sequential nature of the model; usually this happens in cases with flat probability distributions over a number of possible characters following the already predicted character sequence. A large portion of such errors includes incorrect spellings of proper nouns (*proper noun err*): *Jersualm*, *Mconal* instead of *Jerusalem*, *McDonal*. Finally, one prominent group of errors is that of incorrect digit sequences. MorphMlp does not make these mistakes, because it uses a heuristic: OOV lemmas are copied verbatim as predictions of the surface forms.

The majority of erroneous cases for MorphRnnHard model constitute the group of alternative forms. Compared to other models, there are considerably fewer cases of predicting non-existent forms (*allergys*, *goining*). The *wrong form* error type is mainly represented by incorrect predictions of verb forms: *sing*, *got*, *are* instead of *sung*, *gotten*, *'m*.

For our subsequent experiments on the full pipeline we chose the best model, MorphRnnHard. The results of the error analysis suggest that there is still a large room for improvement of the morphological inflection generation component. A principled approach to handling unseen tokens and a way to constrain the predictions to well-formed outputs would be interesting directions to investigate further.

6.5.2 Syntactic component

In order to evaluate the syntactic ordering module, we conducted two experiments.

Experiment 1 First, we performed a preliminary study in which we tried to validate our hypothesis and answer the following question: *Is it possible to accurately predict the relative position of a dependent (child) with respect to its head?* Table 6.5 shows the distribution of left/right target labels in the training data and the accuracy of predicting the node's relative position with our system's binary classifier, both for head-child and sibling node pairs. The latter pairs are relations between sibling nodes in the respective subtree, since at each prediction step the system operates on dependency subtrees of depth one. Note that modeling such relations is a harder task, since siblings in dependency trees do not directly share any grammatical information. However, the surrounding context seems to be enough to make high-accuracy predictions, which supports our hypothesis.

	Language									
	ar	cs	en	es	fi	fr	it	nl	pt	ru
L/R labels (%)	32/68	57/43	62/38	56/44	57/43	57/43	57/43	61/39	56/44	47/53
Pred. Acc. (%)										
head-child	96.27	90.26	96.14	92.67	89.21	95.14	94.26	91.51	97.29	92.85
sibling	82.81	82.78	87.64	83.85	81.43	85.12	84.98	82.52	85.6	85.56

Table 6.5: The distribution of left/right labels in the training data and the accuracy of predicting a node's relative position with the binary classifier. Two cases are considered: predicting the position of a child node (*head-child*), and a sibling (*sibling*).

Experiment 2 The second experiment was supposed to estimate the performance of BINLIN compared to an ideal syntactic ordering module, i.e. an upper bound on its performance. We trained the syntactic ordering component on the training set of the English UD treebank, and performed its automatic metric evaluation by computing BLEU, NIST and i-EDIST scores between the reference sentences and system outputs on the development set. Note that the outputs of BINLIN and upper bound in this experiment contain ordered lemmas, not surface forms, while the references are correctly ordered sequences of inflected surface forms, as given in the CONLL file.

	BLEU	i-EDIST	NIST
BINLIN-syn	51.15	64.78	10.82
Upper bound (oracle)	65.31	85.52	11.38

Table 6.6: BINLIN's performance at the syntactic ordering step, and its upper bound. Computed on the English portion of the SR'18 development set.

Table 6.6 shows reassuring results: the proposed algorithm is reasonably close to the upper bound, confirming our results in the first experiment.

6.5.3 Full Pipeline

We further add the morphological inflection component and evaluate the full pipeline on the SR'18 test data. Before showing the results, we would like to briefly describe the three other best-performing systems that achieved comparable results in SR'18.

6.5.3.1 Compared Approaches

OSU Similarly to BINLIN, the OSU system by King and White (2018) is also a pipeline system. However, it performs the steps in the reverse order. First, it employs a seq2seq model to convert the lemmas to inflected word forms using the grammatical features in the UD data (the *FEAT* column). In the second step, the inflected words are incrementally linearized by a global linear model which relies on the supplied syntactic dependencies and grammatical features. In a nutshell, the first

component is similar to our morphological subsystem, but uses a soft attention mechanism. The second component is very similar to the global linear model of Puduppully et al. (2016).

TILBURG The TILBURG system by Castro Ferreira et al. (2018) also shares some similarities to our approach. The authors use a maximum entropy classifier to decide which first-order child nodes are most likely to be before and after the head nodes, which is somewhat similar to the process of how binary subtrees are being constructed by BINLIN. The node groups are then re-ordered by a merge-sort–style algorithm which uses a separate classifier. The resultant sequence is then post-processed with a lexicon lookup to realize the lemmas to surface forms. Finally, the sequence is fed to a phrase-based machine translation system that finishes the realization process by translating from a partially-realized string to a sentence.

ADAPT The ADAPT system by Elder and Hokamp (2018) approaches the task in an end-to-end manner, akin to neural machine translation systems. First, they prepare the *source* data by flattening the input dependency tree: they re-construct the original parse tree from the dependency features, and perform a depth-first search to sort and reorder the lemmas, thus obtaining a sequence of lemmas. The authors treat the original sentence as the *target* text and train the system end-to-end. The main finding of the authors is that the training dataset size is one of the major obstacles preventing current seq2seq models from doing well at NLG from structured inputs. The authors show that data augmentation boosts the performance of seq2seq models. Unlike us, however, the authors had to use an external resource which is almost 50 times larger than the original SR'18 data.

6.5.3.2 Results

Table 6.7 shows the metric scores achieved by BINLIN, and the three aforementioned systems. BINLIN was trained ten times with different random seeds; we report both the mean scores and standard deviation.

The proposed modifications clearly bridge the gap between BINLIN and the rest of the participating systems. BINLIN was the best-performing system for five out of ten languages, and on the remaining languages it showed comparable performance.

6.5.4 Error Analysis

In order to better understand the most common errors made by the BINLIN system, we manually examined its predictions on the development set. We were focusing predominantly on the syntactic ordering component. In what follows we describe the most prominent error types.

Punctuation Generally speaking, the position of punctuation marks is determined not by a specific dependency relation, but rather by discourse-level characteristics of the sentence, since their primary goal is to help the reader interpret the text by means of delimiting the content, dividing it

		Language				
		ar	cs	en	es	fi
BLEU	ADAPT	-	-	**69.14**	-	-
	OSU	25.65	-	66.33	**65.31**	**37.52**
	TILBURG	-	-	55.29	49.47	-
	BINLIN	**29.07** ± 0.20	**54.49** ± 0.26	64.70 ± 0.76	63.86 ± 0.50	37.38 ± 0.59
NIST	ADAPT	-	-	12.02	-	-
	OSU	7.15	-	12.02	**12.74**	9.56
	TILBURG	-	-	10.85	11.11	-
	BINLIN	**7.53** ± 0.01	**13.65** ± 0.03	**12.03** ± 0.05	12.59 ± 0.03	**9.83** ± 0.06
i-EDIST	ADAPT	-	-	**80.42**	-	-
	OSU	34.37	-	68.59	**59.75**	47.99
	TILBURG	-	-	60.32	48.47	-
	BINLIN	**38.11** ± 0.60	**54.00** ± 0.34	66.77 ± 0.77	59.00 ± 0.46	**49.58** ± 0.65

(a) Languages: Arabic, Czech, English, Spanish, Finnish

		Language				
		fr	it	nl	pt	ru
BLEU	ADAPT	-	-	-	-	-
	OSU	38.24	-	25.52	-	-
	TILBURG	**52.03**	**44.46**	32.28	30.82	-
	BINLIN	43.40 ± 1.33	41.17 ± 0.51	**50.28** ± 0.53	**49.01** ± 0.60	**62.88** ± 0.36
NIST	ADAPT	-	-	-	-	-
	OSU	8.0	-	7.33	-	-
	TILBURG	**9.85**	**9.11**	8.04	7.55	-
	BINLIN	8.78 ± 0.11	8.63 ± 0.03	**10.30** ± 0.02	**9.21** ± 0.05	**14.43** ± 0.02
i-EDIST	ADAPT	-	-	-	-	-
	OSU	44.84	-	34.22	-	-
	TILBURG	**51.16**	46.67	37.20	40.75	-
	BINLIN	45.17 ± 1.10	**48.50** ± 0.86	**52.50** ± 0.65	**59.70** ± 0.78	**62.80** ± 0.54

(b) Languages: French, Italian, Dutch, Portuguese, Russian

Table 6.7: Final results computed on the SR'18 test data. BINLIN results include mean scores and standard deviation (scores averaged across ten models trained with different random seed values). Cells with dashes denote cases for which the respective systems have not submitted any output in the shared task.

into easy-to-process pieces. Oftentimes there are lexical markers (*so*, *because*, *although*) which signal that, for example, a comma should be inserted before or after a phrase:

- *I like chocolate, because it is sweet.*
- *Bryan, you're in, right?*

However, in UD annotation punctuation marks are considered as dependents of the subtree root. The binary classifier fails to encode discourse information, since it mainly looks for local patterns in head-dependent relations. A more global technique of input encoding might alleviate this issue.

Contractions Spanish, Czech, French, Portuguese, Arabic and Italian treebanks contain annotation of multi-word expressions (MWE). Table 6.8 shows the number of unique MWE encountered in the training portion of the UD treebanks.

Language					
ar	pt	it	es	cs	fr
3 010	447	361	300	18	12

Table 6.8: Number of multi-word expressions (MWE) in UD treebanks for the languages included in the SR'18 task (only languages with MWE are shown).

The most common case marked as MWE in the UD treebanks is that of contractions which occur when two adjacent words are merged into one. For example, in French the article *les* contracts with the preposition *à* into a compound article *aux*. English UD annotation does not contain contractions, which is why when developing BinLin we did not encounter this issue.

Our system predicts the relative position of the contraction elements and attempts to conjugate them separately, but does not perform token merging. The following is an example of a contraction in French:

- **Wrong**: *Un autel à Jupiter est érigé à l'emplacement de le Temple.*
- **Correct**: *Un autel à Jupiter est érigé à l'emplacement du Temple.*

The first line is what BinLin would predict; the second is what the correct output should be. We suspect that this is the main reason for the performance gap that exists between BinLin and the best-performing approaches on Spanish, French and Italian data. Interestingly, the authors of the Tilburg system also mentioned that their system could not properly handle the contractions, which lowered the performance of their system, especially on the Portuguese data.

A possible remedy to this limitation could be modeling syntactic ordering and morphological inflection jointly, but we did not explore this direction further. As a quick fix, we added a post-processing step to the outputs of our system,[34] whereby we stitch adjacent contraction items into

[34] For the syntactic component that we trained with ten different random seeds, we chose one variant randomly.

one token. We focused on Czech, French and Italian treebanks, since these languages have very simple contraction cases which we extracted without any knowledge of the respective grammar rules (see Appendix B.1).

Table 6.9 shows an improvement over all three languages, which suggests that this is indeed a promising direction to investigate in detail, and a more principled treatment of contractions would boost the performance of the system even further.

Metric	System	Language		
		cs	fr	it
BLEU	BINLIN	54.50	43.90	40.84
	+ MWE	**54.78**	**49.26**	**52.11**
NIST	BINLIN	13.63	8.85	8.61
	+ MWE	**13.67**	**9.46**	**9.78**
i-EDIST	BINLIN	54.00	46.20	47.57
	+ MWE	**54.07**	**49.45**	**53.25**

Table 6.9: The results of adding a post-processing step of merging MWE tokens for Czech, French and Italian SR'18 test data. + *MWE* denotes the system variant with MWE treatment.

6.5.5 Possible Extensions

Easy-first linearization We have already mentioned that BINLIN is sensitive to the order of processing of the child and sibling nodes. We hypothesize that an initial step of ranking the nodes to determine which ones are most likely to start the sequence would prune the unnecessary and, perhaps, harmful comparisons of, for instance, punctuation with articles. A principled approach would be to define an adaptive model which encodes some notion of processing preference: given a set of tokens, the system should first make predictions it is most confident about, similar to easy-first dependency parsing algorithm (Goldberg and Elhadad, 2010) or imitation learning methods (Lampouras and Vlachos, 2016).

Non-projective dependencies One significant limitation of the proposed method is its inability to handle non-projective dependencies. This is a simplification decision we made when designing the algorithm: at each point we assume that the perfect token order can be retrieved by recursively ordering head-children subtrees, which excludes long-range crossing dependencies from consideration. By doing so we aggressively prune the search space and simplify the inference procedure, but also rule out a smaller class of more complex constructions. This might not be a problem for the English UD data, which has a small number of non-projective dependencies. However, according to the empirical study of Nivre (2006), almost 25 % of the sentences in the Prague Dependency Treebank of Czech (Böhmová, Alena and Hajič, Jan and Hajičová, Eva and Hladká, Barbora, 2003), and more than 15 % in the Danish Dependency Treebank (Kromann, 2003) contain non-projective dependencies. This implies that for multi-lingual surface realization our assumption could be too strong.

6.6 Chapter Summary

In this chapter we approached the problem of designing more reliable NLG systems. We argued that modern neural approaches suffer from the lack of interpretability, mainly caused by the complexity of their architecture. Their black-box behavior makes it hard to analyze the performance or propose further improvements.

We further argued that one way to approach this problem is by means of "separation of concerns": in some tasks it is possible for a human to encode the problem-solving logic in a decision-making algorithm, while the actual decisions can be outsourced to a data-driven system.

In order to test this idea in the NLG domain, we took part in the 2018 Multilingual Surface Realization Workshop. We analyzed the shared task data and concluded that popular seq2seq approaches would likely fail, because the training corpora provided for the task were not big enough for training such models. In addition to that, some of the treebanks represented morphologically rich languages, which exacerbated the data scarcity issue due to large vocabulary sizes. This served as an additional evidence that exploring other approaches might be a better way to solve the task at hand.

Based on task specifications, we proposed a hybrid method of surface realization, which addresses the aforementioned issues in two ways. First, it separates the syntactic ordering and morphological inflection decisions, thus reducing the space of possible output candidates. Second, the syntactic ordering component performs dependency tree linearization by following an algorithm that is designed by a human; only local decisions are performed by a neural network classifier.

The proposed method has a number of attractive properties. First, it guarantees that the resulting sequence of tokens is a subset of the input, meaning that there is less possibility of "hallucinating" new content during generation. Introducing new content is a problem, because it would mean that the system output is no longer faithful to the system input. This is a common issue with most seq2seq models (Nie et al., 2019; Filippova, 2020; Fadaee and Monz, 2020), and our approach has a clear advantage in this respect.

Second, the proposed method is more intuitive and interpretable. It supports separate analysis of the syntactic ordering and morphological inflection steps of the surface linearization process. From a research perspective, this offers greater control over the problem-solving procedure.

The empirical evidence showed that the syntactic ordering technique which we proposed is capable of performing accurate dependency tree linearization. Our system performed comparably to the rest of the systems; on half of the languages our system outperformed the rest of the participants.

CHAPTER 7
Conclusion

This chapter summarizes the findings of our thesis and outlines potential future research directions.

7.1 Summary of Findings and Contributions

In this thesis we introduced the *Sanity Poligon*, a new methodological framework, to approach the task of NLG from a holistic point of view. We decomposed the problem specification into five components which are essential for any NLP study: task, data, evaluation criteria, metrics and human evaluation experiment. We further defined a set of core principles which govern the relations between these components. The framework is envisioned as a guide which one can use during the design and analysis of NLG experiments.

In the subsequent chapters we described a series of NLG experiments that demonstrate the efficacy of the framework in approaching the research questions we outlined in the introduction chapter.

Chapter 3 We approached *RQ1* by introducing SPF, a conceptual framework for the design and analysis of NLG studies. We motivated the need for such a framework by the existent fragmentation problem of the NLG field, largely caused by the lack of common standards in task specification. The proposed framework consists of a set of principles and recommendations grouped by relations between the main components of an NLP experiment. We summarized the principles in a checklist form which we apply in the subsequent chapters.

Chapter 4 In this chapter we presented our experiments in the data-to-text generation task, sought to put the designed framework to test. We described the results of our participation in the E2E NLG Challenge, which we viewed as a case study to provide empirical evidence of the utility of SPF. We showed how one can use the framework to detect and address possible evaluation issues, as well as develop new approaches which aim at solving the task at hand, as opposed to obtaining higher metric scores. We were able to show how to use the framework to approach *RQ2*: a template-based system was designed according to task specifications. We

provided empirical evidence that it offers a good cost–quality trade-off, while being more reliable than its data-driven competitor.

Chapter 5 This chapter showcased how SPF can be used to critically analyze evaluation setups, revealing weak spots of system assessment protocols. We presented a sentence compression case study in which we used the SPF principles to detect an evaluation discrepancy type which prior work seems to have overlooked. We drew connections with the study described in Chapter 4 and provided a possible explanation for this phenomenon: the data is noisy and errors find their way into system predictions, but they are hard to detect, because the overall quality of the systems trained on this data is high. Focusing on the worst-case performance is the way we explored in our experiments, and it proved to be a correct strategy for this scenario.

Chapter 6 We described another solution to *RQ2*. We argued that modern neural approaches suffer from the lack of interpretability, caused by the complexity of their architecture. Their black-box behavior makes it hard to rely on them in practice, since their performance depends on factors which are hard to envision. We proposed to approach this problem by means of separation of the problem-solving logic from the decision-making process; by letting a data-driven system make localized decisions. We used surface realization as an example task and described a hybrid system which addresses the aforementioned issues. The empirical evidence showed that the proposed system achieves state-of-the-art performance, while offering a good trade-off between performance gains and interpretability.

7.2 Future Research Directions

As we mentioned before, the number of research papers and scientific events dedicated to NLG problems is constantly growing. The attention that the field is attaining is outpacing the development of the methodological basis, necessary to conduct solid research. We view this thesis is a humble contribution in this respect and hope that its results will be of use to the upcoming generations of NLG researchers. There are several ways in which the proposed framework and techniques can be improved.

7.2.1 Extending SPF

Component improvements The developed framework is not complete, since it was not meant to exhaustively cover all interactions between the NLG components. We barely scratched the surface of an important question of human experiment design, mainly because we did not perform human evaluation experiments in the course of the thesis study. Another reason is the lack of information about the details of human experiments reported in the literature.

Relation Intersection One way to extend this framework is by connecting principles between each other, although that would complicate the framework design. For example, the analysis of

differences between the assumptions made during metric development and the ones made when planning human evaluation experiments poses a great interest, since both metric and human evaluations are developed as a way to assess system performance. Such analysis could provide more explanation to the issue of low correlation between human judgments of textual quality and automatic metric scores.

Presentation improvements We developed SPF as a set of recommendations, but a checklist form was a natural way to formalize it. However, it is not the most usable way of assisting someone in a study analysis. We believe that something closer to a *decision-tree* would be more practical and usable. In this way, a researcher would not need to go through the relations and principles to check the relevant ones. One could simply start at the root of the tree and answer the questions to find the potentially problematic ones.

Application Potential It would be worthwhile to extend the scope of the framework to other NLP tasks. As we mentioned before, it is not specific to text generation. It would be interesting to see what kind of evidence (or counter-facts) can be found in a cross-application setting.

7.2.2 Extrinsic Studies Inclusion

As we mentioned before, extrinsic task-based evaluation results serve as the most convincing evidence of the usability of a technique in NLG. However, it seems that this type of evaluation is losing its popularity: Van der Lee et al. (2019) report an ongoing dramatic decrease in the number of studies with extrinsic evaluation experiments (a drop from around 25 % in 2005–2014 to 3 % in 2018). There might be many reasons for this: high deployment costs, application specificity of the NLG systems that can be used in real-world scenarios, short life-cycle of modern data-driven techniques, etc.

Nevertheless, the cost–quality trade-off issue that we mentioned in Section 4.6.2 raises a bar for NLG techniques. System assessment in a real-world scenario is expensive and hard to reproduce, which means it is hard to generalize the results from extrinsic evaluation studies. However, they bring additional value, as opposed to a dry comparison with metric scores reported in some prior work, and it would be very interesting to include this value into the developed framework. For example, extrinsic studies have a potential to reveal unexpected behavior of the users, which can be accounted for when planning evaluation experiments in NLG (Carter, 1996). Additionally, such studies inform the community about the requirements that an NLG system needs to meet to be useful in real world. For instance, evaluation of the utility of NLG systems in summarizing clinical data about babies in neonatal intensive care units (Portet et al., 2009; Mahamood and Reiter, 2011; Hunter et al., 2012) has shown that such systems lack in maintainability and pose challenges from a regulatory perspective, when being deployed.[35] This information is important, but cannot be obtained during intrinsic system evaluation.

[35] https://ehudreiter.com/2019/07/08/why-isnt-research-software-used/

7.2.3 Sentence Compression Experiments Extension

The evaluation discrepancy we discovered in our sentence compression experiments is thought-provoking. Perhaps, we would not have been able to find it, if it were not for our unusual experimental setup: for the error analysis we did not sample instances randomly, but biased the results in favor of the higher-scoring model which, however, showed worse results in a manual error analysis.

Further investigation is needed to make stronger claims about our observations. The study's findings should be confirmed for other datasets and, perhaps, tasks.

7.2.4 Hybrid NLG Systems

It would be interesting to conduct a more in-depth study on the limitations of end-to-end systems that can be solved by hybrid systems. It is clear that the reason why our approach performed better than, for example, ADAPT (with an equal amount of data), is a better sample complexity of BINLIN. However, it is not possible to develop a designated algorithm for each and every NLP task. It works for dependency parsing or surface realization, but is not possible for more high-level problems, like machine translation or question answering, because such problems assume a very large space of output candidates (Hirschmann et al., 2016). Determining task specifications which permit the development of hybrid approaches is an interesting research question worth exploring in future.

In our surface realization experiments, we decomposed the prediction into two separate stages of syntactic ordering and word inflection. We argued that this offers a greater control over the generation process. However, joint morphological inflection and syntactic ordering could potentially bring additional performance gains. How to do that without losing the inference transparency is a promising research topic to examine.

Appendix

A E2E NLG Challenge Experiments

A.1 Manual Data Analysis Results

Manual analysis of the training set in the E2E NLG Challenge revealed certain annotation problems. Below we provide sentence IDs of the instances which we considered as containing errors.

Modified contents 4136, 34141, 32915, 35936, 6152, 2005, 1463, 14529, 14339, 21804, 25779, 11049.

Dropped contents 4136 (*price and food*), 17455 (*familyFriendly*), 38742 (*area*), 1463 (*customerRating*), 27664 (*food, priceRange*), 19882 (*priceRange*), 25779 (*customerRating, area*), 40293 (*food*), 807 (*familyFriendly*), 15216 (*familyFriendly*).

Ungrammatical 39810, 1904, 36233, 33671, 28698;

- tweet-style descriptions: 19425, 38875, 14873;
- word attachment problems, bad phrasing: 15276, 30950, 31973, 36468, 12442, 18623, 25773.

Punctuation errors 6152, 14993, 38742, 14339, 35967, 25795.

Questionable realization

- *customerRating*: 8806, 24983, 29189;
- *familyFriendly*: 20220, 10729;
- *priceRange*: 29189;
- *style*: 14993, 28362, 32494.

A.2 Example Predictions

The majority of predictions from all three systems are grammatically correct. Here is a random sample:

- **Input:** name[Cocum], eatType[coffee shop], food[English], priceRange[moderate], customerRating[3 out of 5], familyFriendly[no]
- **TGen:** Cocum is a moderately priced coffee shop serving english food. It has a customer rating of 3 out of 5 and is not kid friendly.
- **Model-D:** Cocum is a moderately priced coffee shop serving English food . It is not kid friendly and has a customer rating of 3 out of 5 .
- **Model-T:** Cocum is a coffee shop which serves English food in the moderate price range. It has a customer rating of 3 out of 5. It is not family friendly.

To illustrate common prediction mistakes (regardless of the system), we have hand-picked some samples. Full predictions by all models can be found in the GitHub repository.

A.2.1 Bad Grammar

- **Input:** name[The Wrestlers], eatType[coffee shop], food[Chinese], priceRange[less than £20], area[riverside], familyFriendly[no], near[Raja Indian Cuisine]
- **TGen:** The Wrestlers is a coffee shop providing chinese food in the less than £20 price range. It is located in the riverside. It is near Raja Indian Cuisine. Its no children feel allowed.
- **Model-D:** The Wrestlers is a coffee shop providing Chinese food in the less than £20 price range . It is located in the riverside . It is near Raja Indian Cuisine .
- **Model-T:** The Wrestlers is a coffee shop which serves Chinese food in the price range of less than £20. It is located in the riverside area, near Raja Indian Cuisine. It is not family friendly.

- **Input:** name[Wildwood], eatType[coffee shop], food[English], priceRange[high], customerRating[1 out of 5], near[Ranch]
- **TGen:** Wildwood is a high priced coffee shop that serves english food. It is located near Ranch and has a customer rating of 1 out of 5.
- **Model-D:** Wildwood is a coffee shop near Ranch , serving English food , has a high price range and a customer rating of 1 out of 5 .
- **Model-T:** Wildwood is a coffee shop which serves English food in the high price range. It has a customer rating of 1 out of 5 and is located near Ranch.

A.2.2 Modified and/or Dropped Contents

- **Input:** name[The Golden Palace], eatType[coffee shop], food[English], priceRange[high], customerRating[average], area[riverside]

- **TGen:** The Golden Palace is a high priced coffee shop serving english food in the riverside area. It has an average customer rating.
- **Model-D:** The Golden Palace is a high priced coffee shop located in the city centre .
- **Model-T:** The Golden Palace is a coffee shop which serves English food in the high price range. It has an average customer rating and is located in the riverside area.

A.2.3 Dropped Contents

- **Input:** name[The Eagle], eatType[coffee shop], food[English], priceRange[moderate], customerRating[1 out of 5], area[riverside], familyFriendly[yes], near[Burger King]
- **TGen:** The Eagle is a moderately priced coffee shop that serves english food. It is located in the riverside area near Burger King. It is kid friendly and has a customer rating of 1 out of 5.
- **Model-D:** The Eagle is a moderately priced coffee shop located near Burger King .
- **Model-T:** The Eagle is a family-friendly coffee shop which serves English food in the moderate price range. It has a customer rating of 1 out of 5 and is located in the riverside area, near Burger King.

B Surface Realization Experiments

B.1 Contraction Rules

The following tables contain the contraction rules we used as a post-processing step described in Section 6.5.4. The rules were created by extracting lines with contractions in the UD CoNLL files and analyzing the contraction patterns.

à le → au	de lequel → duquel
à les → aux	de lesquels → desquels
à lequel → auquel	de lesquelles → desquelles
à lesquels → auxquels	en les → ès
à lesquelles → auxquelles	vois ci → voici
de le → du	vois là → voilà
de les → des	

(a) French contraction rules

di il → del	a l'→ all'
di lo → dello	a le → alle
di la → della	a i → ai
di l' → dell'	a gli → agli
di i → dei	su il → sul
di gli → degli	su la → sulla
di le → delle	su lo → sullo
a il → al	su gli → sugli
a lo → allo	con il → col
a la → alla	con i → coi

(b) Italian contraction rules

aby by → aby	když by → kdyby
Aby by → Aby	Když by → Kdyby

(c) Czech contraction rules

Table B.1: Contraction rules for French, Italian and Czech, used in the MWE experiments (Section 6.5.4).

List of Figures

List of Tables

Bibliography

Aharoni, R. and Goldberg, Y. (2017). Morphological Inflection Generation with Hard Monotonic Attention. In *Proceedings of the 55th Annual Meeting of the Association for Computational Linguistics*, pages 2004–2015, Vancouver, Canada.

Ahlberg, M., Forsberg, M., and Hulden, M. (2015). Paradigm Classification in Supervised Learning of Morphology. In *Proceedings of the 2015 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 1024–1029, Denver, Colorado, USA.

Amidei, J., Piwek, P., and Willis, A. (2018). Evaluation Methodologies in Automatic Question Generation 2013–2018. In *Proceedings of the 11th International Conference on Natural Language Generation*, pages 307–317, Tilburg University, Netherlands.

Amidei, J., Piwek, P., and Willis, A. (2019). The Use of Rating and Likert Scales in Natural Language Generation Human Evaluation Tasks: a Review and Some Recommendations. In *Proceedings of the 12th International Conference on Natural Language Generation*, pages 397–402, Tokyo, Japan.

Anderson, P., Fernando, B., Johnson, M., and Gould, S. (2016). SPICE: Semantic Propositional Image Caption Evaluation. In *Proceedings of the 2016 European Conference on Computer Vision*, pages 382–398, Amsterdam, Netherlands.

Angeli, G., Johnson Premkumar, M. J., and Manning, C. D. (2015). Leveraging Linguistic Structure For Open Domain Information Extraction. In *Proceedings of the 53rd Annual Meeting of the Association for Computational Linguistics and the 7th International Joint Conference on Natural Language Processing*, pages 344–354, Beijing, China.

Ariely, D. (2009). *Predictably Irrational: the Hidden Forces That Shape Our Decisions*. Harper, New York, New York, USA.

Aroyo, L. and Welty, C. (2015). Truth Is a Lie: Crowd Truth and the Seven Myths of Human Annotation. *AI Magazine*, 36(1):15–24.

Attardi, G. (2006). Experiments with a Multilanguage Non-projective Dependency Parser. In *Proceedings of the 10th Conference on Computational Natural Language Learning*, pages 166–170, New York, New York, USA.

Babych, B. and Hartley, A. (2008). Sensitivity of Automated MT Evaluation Metrics on Higher Quality MT Output: BLEU vs Task-based Evaluation Methods. In *Proceedings of the 6th International Conference on Language Resources and Evaluation*, pages 2133–2136, Marrakech, Morocco.

Bahdanau, D., Cho, K., and Bengio, Y. (2015). Neural Machine Translation by Jointly Learning to Align and Translate. In *Proceedings of the 3rd International Conference on Learning Representations (ICLR 2015)*, San Diego, California, USA.

Banarescu, L., Bonial, C., Cai, S., Georgescu, M., Griffitt, K., Hermjakob, U., Knight, K., Koehn, P., Palmer, M., and Schneider, N. (2013). Abstract Meaning Representation for Sembanking. In *Proceedings of the 7th Linguistic Annotation Workshop and Interoperability with Discourse*, pages 178–186, Sofia, Bulgaria.

Banerjee, S. and Lavie, A. (2005). METEOR: an Automatic Metric for MT Evaluation with Improved Correlation with Human Judgments. In *Proceedings of the ACL Workshop on Intrinsic and Extrinsic Evaluation Measures for Machine Translation and/or Summarization*, pages 65–72, Ann Arbor, Michigan, USA.

Bangalore, S. and Rambow, O. (2000). Exploiting a Probabilistic Hierarchical Model for Generation. In *Proceedings of the 18th Conference on Computational Linguistics*, pages 42–48, Saarbrücken, Germany.

Barzilay, R., Elhadad, N., and McKeown, K. R. (2002). Inferring Strategies for Sentence Ordering in Multidocument News Summarization. *Journal of Artificial Intelligence Research*, 17:35–55.

Barzilay, R. and Lapata, M. (2005). Collective Content Selection for Concept-to-text Generation. In *Proceedings of Human Language Technology Conference and Conference on Empirical Methods in Natural Language Processing*, pages 331–338, Vancouver, Canada.

Barzilay, R. and Lapata, M. (2006). Aggregation via Set Partitioning for Natural Language Generation. In *Proceedings of the 2006 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 359–366, New York, New York, USA.

Barzilay, R. and Lee, L. (2004). Catching the Drift: Probabilistic Content Models, with Applications to Generation and Summarization. In *Proceedings of the 2004 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 113–120, Boston, Massachusetts, USA.

Barzilay, R. and McKeown, K. R. (2005). Sentence Fusion for Multidocument News Summarization. *Computational Linguistics*, 31(3):297–328.

Bateman, J. A. (1992). Towards Meaning-based Machine Translation: Using Abstractions from Text Generation for Preserving Meaning. *Machine Translation*, 7(1/2):5–40.

Belz, A. (2008). Automatic Generation of Weather Forecast Texts Using Comprehensive Probabilistic Generation-space Models. *Natural Language Engineering*, 14(4):431–455.

Belz, A. (2009). Last Words: That's Nice ... What Can You Do with It? *Computational Linguistics*, 35(1):111–118.

Belz, A. and Gatt, A. (2008). Intrinsic vs. Extrinsic Evaluation Measures for Referring Expression Generation. In *Proceedings of the 46th Annual Meeting of the Association for Computational Linguistics on Human Language Technologies: Short Papers*, pages 197–200, Columbus, Ohio, USA.

Belz, A. and Kow, E. (2010). Comparing Rating Scales and Preference Judgements in Language Evaluation. In *Proceedings of the 6th International Natural Language Generation Conference*, pages 7–15, Trim, Co. Meath, Ireland.

Belz, A. and Kow, E. (2011). Discrete vs. Continuous Rating Scales for Language Evaluation in NLP. In *Proceedings of the 49th Annual Meeting of the Association for Computational Linguistics: Human Language Technologies*, pages 230–235, Portland, Oregon, USA.

Belz, A., Kow, E., Viethen, J., and Gatt, A. (2010). *Generating Referring Expressions in Context: the GREC Task Evaluation Challenges*, pages 294–327. Springer Berlin Heidelberg.

Belz, A. and Reiter, E. (2006). Comparing Automatic and Human Evaluation of NLG Systems. In *Proceedings of the 11th Conference of the European Chapter of the Association for Computational Linguistics*, pages 313–320, Trento, Italy.

Belz, A., White, M., Espinosa, D., Kow, E., Hogan, D., and Stent, A. (2011). The 1st Surface Realisation Shared Task: Overview and Evaluation Results. In *Proceedings of the Generation Challenges Session at the 13th European Workshop on Natural Language Generation*, pages 217–226, Nancy, France.

Bengio, S., Vinyals, O., Jaitly, N., and Shazeer, N. (2015). Scheduled Sampling for Sequence Prediction with Recurrent Neural Networks. In *Proceedings of the 28th International Conference on Neural Information Processing Systems*, pages 1171–1179, Montréal, Canada.

Berk, R. A. (1983). An Introduction to Sample Selection Bias in Sociological Data. *American Sociological Review*, 48(3):386–398.

Black, R., Reddington, J., Reiter, E., Tintarev, N., and Waller, A. (2010). Using NLG and Sensors to Support Personal Narrative for Children with Complex Communication Needs. In *Proceedings of the NAACL HLT 2010 Workshop on Speech and Language Processing for Assistive Technologies*, pages 1–9, Los Angeles, California, USA.

Böhmová, Alena and Hajič, Jan and Hajičová, Eva and Hladká, Barbora (2003). *The Prague Dependency Treebank: a Three-level Annotation Scenario*, chapter 7, pages 103–127. Kluwer Academic Publishers.

Bohnet, B., Wanner, L., Mille, S., and Burga, A. (2010). Broad Coverage Multilingual Deep Sentence Generation with a Stochastic Multi-level Realizer. In *Proceedings of the 23rd International Conference on Computational Linguistics*, pages 98–106, Beijing, China.

Bottou, L. (1999). *On-line Learning and Stochastic Approximations*, pages 9–42. Cambridge University Press, USA.

Boyer, K. E., Ha, E. Y., Phillips, R., Wallis, M. D., Vouk, M. A., and Lester, J. C. (2009). Inferring Tutorial Dialogue Structure with Hidden Markov Modeling. In *Proceedings of the 4th Workshop on Innovative Use of NLP for Building Educational Applications*, pages 19–26, Boulder, Colorado, USA.

Bozinovski, S. and Fulgosi, A. (1976). The Influence of Pattern Similarity and Transfer Learning upon the Training of a Base Perceptron B2 (Original in Croatian: Utjecaj Slicnosti Likova I Transfera Ucenja Na Obucavanje Baznog Perceptrona B2). In *Proceedings of Symposium Informatica*, volume 3-121-5, Bled, Slovenia.

Britz, D., Goldie, A., Luong, M.-T., and Le, Q. (2017). Massive Exploration of Neural Machine Translation Architectures. In *Proceedings of the 2017 Conference on Empirical Methods in Natural Language Processing*, pages 1442–1451, Copenhagen, Denmark.

Brown, T., Mann, B., Ryder, N., Subbiah, M., Kaplan, J. D., Dhariwal, P., Neelakantan, A., Shyam, P., Sastry, G., Askell, A., Agarwal, S., Herbert-Voss, A., Krueger, G., Henighan, T., Child, R., Ramesh, A., Ziegler, D., Wu, J., Winter, C., Hesse, C., Chen, M., Sigler, E., Litwin, M., Gray, S., Chess, B., Clark, J., Berner, C., McCandlish, S., Radford, A., Sutskever, I., and Amodei, D. (2020). Language Models Are Few-shot Learners.

In *Advances in Neural Information Processing Systems*, volume 33, pages 1877–1901, Montréal, Canada.

Brysbaert, M. (2019). How Many Participants Do We Have to Include in Properly Powered Experiments? A Tutorial of Power Analysis with Reference Tables. *Journal of Cognition*, 2(1):1–38.

Callison-Burch, C. and Dredze, M. (2010). Creating Speech and Language Data with Amazon's Mechanical Turk. In *Proceedings of the NAACL HLT 2010 Workshop on Creating Speech and Language Data with Amazon's Mechanical Turk*, pages 1–12, Los Angeles.

Callison-Burch, C., Fordyce, C., Koehn, P., Monz, C., and Schroeder, J. (2007). (Meta-) Evaluation of Machine Translation. In *Proceedings of the 2nd Workshop on Statistical Machine Translation*, pages 136–158, Prague, Czech Republic.

Callison-Burch, C., Osborne, M., and Koehn, P. (2006). Re-evaluating the Role of BLEU in Machine Translation Research. In *Proceedings of the 11th Conference of the European Chapter of the Association for Computational Linguistics*, pages 249–256, Trento, Italy.

Carenini, G. and Moore, J. D. (2006). Generating and Evaluating Evaluative Arguments. *Artificial Intelligence*, 170(11):925–952.

Carroll, J., Copestake, A., Flickinger, D., and Poznanski, V. (1999). An Efficient Chart Generator for (Semi-)Lexicalist Grammars. In *Proceedings of the 7th European Workshop on Natural Language Generation*, pages 86–95, Toulouse, France.

Carter, E. (1996). *Quantitative Analysis of hypertext Generation and Organisation Techniques*. PhD thesis, Department of Artificial Intelligence, The University of Edinburgh, Scotland.

Carvalho, D. V., Pereira, E. M., and Cardoso, J. S. (2019). Machine Learning Interpretability: a Survey on Methods and Metrics. *Electronics*, 8(8).

Castro Ferreira, T., Calixto, I., Wubben, S., and Krahmer, E. (2017). Linguistic realisation as machine translation: Comparing different MT models for AMR-to-text generation. In *Proceedings of the 10th International Conference on Natural Language Generation*, pages 1–10, Santiago de Compostela, Spain.

Castro Ferreira, T., Wubben, S., and Krahmer, E. (2018). Surface Realization Shared Task 2018 (SR18): The Tilburg University Approach. In *Proceedings of the 1st Workshop on Multilingual Surface Realisation*, pages 35–38, Melbourne, Australia.

Chaganty, A., Mussmann, S., and Liang, P. (2018). the Price of Debiasing Automatic Metrics in Natural Language Evalaution. In *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics*, pages 643–653, Melbourne, Australia.

Chen, D., Fisch, A., Weston, J., and Bordes, A. (2017a). Reading Wikipedia to Answer Open-domain Questions. In *Proceedings of the 55th Annual Meeting of the Association for Computational Linguistics*, pages 1870–1879, Vancouver, Canada.

Chen, D. L. and Mooney, R. J. (2008). Learning to Sportscast: a Test of Grounded Language Acquisition. In *Proceedings of the 25th International Conference on Machine Learning*, pages 128–135, Helsinki, Finland.

Chen, Q., Hu, Q., Huang, J. X., He, L., and An, W. (2017b). Enhancing Recurrent Neural Networks with Positional Attention for Question Answering. In *Proceedings of the 40th International ACM SIGIR Conference on Research and Development in Information Retrieval*, pages 993–996, Shinjuku, Tokyo, Japan.

Chen, Y.-C., Gan, Z., Cheng, Y., Liu, J., and Liu, J. (2020a). Distilling Knowledge Learned in BERT for Text Generation. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 7893–7905, Online.

Chen, Z., Eavani, H., Chen, W., Liu, Y., and Wang, W. Y. (2020b). Few-shot NLG with Pre-trained Language Model. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 183–190, Online.

Cho, K., van Merrienboer, B., Bahdanau, D., and Bengio, Y. (2014a). On the Properties of Neural Machine Translation: Encoder-decoder Approaches. In *Proceedings of SSST-8, the 8th Workshop on Syntax, Semantics and Structure in Statistical Translation*, pages 103–111, Doha, Qatar.

Cho, K., van Merriënboer, B., Gulcehre, C., Bahdanau, D., Bougares, F., Schwenk, H., and Bengio, Y. (2014b). Learning Phrase Representations Using RNN Encoder–decoder for Statistical Machine Translation. In *Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 1724–1734, Doha, Qatar.

Cho, K., van Merrienboer, B., Gulcehre, C., Bahdanau, D., Bougares, F., Schwenk, H., and Bengio, Y. (2014c). Learning Phrase Representations Using RNN Encoder-decoder for Statistical Machine Translation. In *Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing*, pages 1724–1734, Doha, Qatar.

Chorowski, J., Bahdanau, D., Serdyuk, D., Cho, K., and Bengio, Y. (2015). Attention-based Models for Speech Recognition. In *Proceedings of the 28th International Conference on Neural Information Processing Systems*, pages 577–585, Montréal, Canada.

Clark, A. and Tim, I. (2003). Pre-processing Very Noisy Text. In *Proceedings of The 2003 Workshop on Shallow Processing of Large Corpora*, pages 12–22, Lancaster, UK.

Clark, K., Luong, M.-T., Le, Q. V., and Manning, C. D. (2020). Electra: Pretraining Text Encoders as Discriminators Rather than Generators. In *Proceedings of the 8th International Conference on Learning Representations*, Addis Ababa, Ethiopia.

Clinchant, S., Jung, K. W., and Nikoulina, V. (2019). On the Use of BERT for Neural Machine Translation. In *Proceedings of the 3rd Workshop on Neural Generation and Translation*, pages 108–117, Hong Kong.

Coavoux, M., Narayan, S., and Cohen, S. B. (2018). Privacy-preserving Neural Representations of Text. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing*, pages 1–10, Brussels, Belgium.

Cohen, J. (1960). A Coefficient of Agreement for Nominal Scales. *Educational and Psychological Measurement*, 20(1):37–46.

Cohn, T. and Lapata, M. (2008). Sentence Compression Beyond Word Deletion. In *Proceedings of the 22nd International Conference on Computational Linguistics (Coling 2008)*, pages 137–144, Manchester, UK.

Cui, Y., Che, W., Liu, T., Qin, B., Wang, S., and Hu, G. (2019). Cross-lingual Machine Reading Comprehension. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing*, pages 1586–1595, Hong Kong, China.

Cui, Y., Chen, Z., Wei, S., Wang, S., Liu, T., and Hu, G. (2017). Attention-over-attention Neural Networks for Reading Comprehension. In *Proceedings of the 55th Annual Meeting of the Association for Computational Linguistics*, pages 593–602, Vancouver, Canada.

Dai, A. M. and Le, Q. V. (2015). Semi-supervised Sequence Learning. In *Advances in Neural Information Processing Systems*, volume 28, Montréal, Canada.

Dale, R., Geldof, S., and Prost, J.-P. (2003). CORAL: Using Natural Language Generation for Navigational Assistance. In *Proceedings of the 26th Australasian Computer Science Conference*, volume 16, pages 35–44.

Dale, R. and Mellish, C. (1998). Towards the Evaluation of Natural Language Generation. In *Proceedings of the 1st International Conference on Language Resources and Evaluation*, Granada, Spain.

Dale, R. and Reiter, E. (1995). Computational Interpretations of the Gricean Maxims in the Generation of Referring Expressions. *Cognitive Science*, 19:233–263.

Daniel, F., Kucherbaev, P., Cappiello, C., Benatallah, B., and Allahbakhsh, M. (2018). Quality Control in Crowdsourcing: a Survey of Quality Attributes, Assessment Techniques, and Assurance Actions. *ACM Computing Surveys*, 51(1):1–40.

Danilevsky, M., Qian, K., Aharonov, R., Katsis, Y., Kawas, B., and Sen, P. (2020). A Survey of the State of Explainable AI for Natural Language Processing. In *Proceedings of the 1st Conference of the Asia-Pacific Chapter of the Association for Computational Linguistics and the 10th International Joint Conference on Natural Language Processing*, pages 447–459, Suzhou, China.

De Clercq, O. and Hoste, V. (2016). All Mixed up? Finding the Optimal Feature Set for General Readability Prediction and Its Application to English and Dutch. *Computational Linguistics*, 42(3):457–490.

De Marneffe, M.-C., Dozat, T., Silveira, N., Haverinen, K., Ginter, F., Nivre, J., and Manning, C. D. (2014). Universal Stanford Dependencies: a Cross-linguistic Typology. In *Proceedings of the 9th International Conference on Language Resources and Evaluation*, pages 4585–4592, Reykjavik, Iceland.

Devlin, J., Chang, M.-W., Lee, K., and Toutanova, K. (2019). BERT: Pretraining of Deep Bidirectional Transformers for Language Understanding. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 4171–4186, Minneapolis, Minnesota, USA.

Dhingra, B., Liu, H., Yang, Z., Cohen, W., and Salakhutdinov, R. (2017). Gated-attention Readers for Text Comprehension. In *Proceedings of the 55th Annual Meeting of the Association for Computational Linguistics*, pages 1832–1846, Vancouver, Canada.

Di Eugenio, B., Fossati, D., Yu, D., Haller, S., and Glass, M. (2005). Aggregation Improves Learning: Experiments in Natural Language Generation for Intelligent Tutoring Systems. In *Proceedings of the 43rd Annual Meeting of the Association for Computational Linguistics (ACL'05)*, pages 50–57, Ann Arbor, Michigan, USA.

Di Eugenio, B., Glass, M., and Trolio, M. (2002). The DIAG Experiments: Natural Language Generation for Tutoring Systems. In *Proceedings of the 2nd International Conference on Natural Language Generation*, pages 120–127, Harriman, New York, USA.

Do Dinh, E.-L. and Gurevych, I. (2016). Token-level Metaphor Detection Using Neural Networks. In *Proceedings of the 4th Workshop on Metaphor in NLP*, pages 28–33, San Diego, California.

Doddington, G. (2002). Automatic Evaluation of Machine Translation Quality Using N-gram Co-occurrence Statistics. In *Proceedings of the 2nd International Conference on Human Language Technology Research*, pages 138–145, San Diego, California, USA.

Dorr, B., Zajic, D., and Schwartz, R. (2003). Hedge Trimmer: a Parse-and-trim Approach to Headline Generation. In *Proceedings of the HLT-NAACL 03 Text Summarization Workshop*, pages 1–8, Edmonton, Canada.

Dreyer, M. and Eisner, J. (2011). Discovering Morphological Paradigms from Plain Text Using a Dirichlet Process Mixture Model. In *Proceedings of the 2011 Conference on Empirical Methods in Natural Language Processing*, pages 616–627, Edinburgh, Scotland, UK.

Duboue, P. and McKeown, K. (2002). Content Planner Construction via Evolutionary Algorithms and a Corpus-based Fitness Function. In *Proceedings of the International Natural Language Generation Conference*, pages 89–96, Harriman, New York, USA.

Durrett, G. and DeNero, J. (2013). Supervised Learning of Complete Morphological Paradigms. In *Proceedings of the 2013 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 1185–1195, Atlanta, Georgia.

Dušek, O. and Jurčíček, F. (2016). Sequence-to-sequence Generation for Spoken Dialogue via Deep Syntax Trees and Strings. In *Proceedings of the 54th Annual Meeting of the Association for Computational Linguistics*, pages 45–51, Berlin, Germany.

Dušek, O., Novikova, J., and Rieser, V. (2018). Findings of the E2E NLG Challenge. In *Proceedings of the 11th International Conference on Natural Language Generation*, pages 322–328, Tilburg University, Netherlands.

Dušek, O., Novikova, J., and Rieser, V. (2020). Evaluating the State-of-the-art of End-to-end Natural Language Generation: the E2E Nlg Challenge. *Computer Speech and Language*, 59:123–156.

Eger, S., Gao, Y., Peyrard, M., Zhao, W., and Hovy, E., editors (2020). *Proceedings of the 1st Workshop on Evaluation and Comparison of NLP Systems*, Online. Association for Computational Linguistics.

Elazar, Y. and Goldberg, Y. (2018). Adversarial Removal of Demographic Attributes from Text Data. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing*, pages 11–21, Brussels, Belgium.

Elder, H. and Hokamp, C. (2018). Generating High-quality Surface Realizations Using Data Augmentation and Factored Sequence Models. In *Proceedings of the 1st Workshop on Multilingual Surface Realisation*, pages 49–53, Melbourne, Australia.

Elhadad, M. and Robin, J. (1992). Controlling Content Realization with Functional Unification Grammars. In *Proceedings of the 6th International Workshop on Natural Language Generation: Aspects of Automated Natural Language Generation*, pages 89–104, London, UK.

Elliott, D. and Keller, F. (2014). Comparing Automatic Evaluation Measures for Image Description. In *Proceedings of the 52nd Annual Meeting of the Association for Computational Linguistics*, pages 452–457, Baltimore, Maryland, USA.

Espinosa, D., Rajkumar, R., White, M., and Berleant, S. (2010). Further Meta-evaluation of Broad-coverage Surface Realization. In *Proceedings of the 2010 Conference on Empirical Methods in Natural Language Processing*, pages 564–574, Cambridge, MA, USA.

Etzioni, O., Banko, M., Soderland, S., and Weld, D. S. (2008). Open Information Extraction from the Web. *ACM Computing Surveys*, 51(12):68–74.

Fadaee, M. and Monz, C. (2020). The Unreasonable Volatility of Neural Machine Translation Models. In *Proceedings of the 4th Workshop on Neural Generation and Translation*, pages 88–96, Online.

Faruqui, M., Tsvetkov, Y., Neubig, G., and Dyer, C. (2016). Morphological Inflection Generation Using Character Sequence-to-sequence Learning. In *Proceedings of the 2016 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 634–643, San Diego, California, USA.

Filippova, K. (2020). Controlled Hallucinations: Learning to Generate Faithfully from Noisy Data. In *Findings of the Association for Computational Linguistics: EMNLP 2020*, pages 864–870, Online.

Filippova, K., Alfonseca, E., Colmenares, C. A., Kaiser, L., and Vinyals, O. (2015). Sentence Compression by Deletion with LSTMs. In *Proceedings of the 2015 Conference on Empirical Methods in Natural Language Processing*, pages 360–368, Lisbon, Portugal.

Filippova, K. and Altun, Y. (2013). Overcoming the Lack of Parallel Data in Sentence Compression. In *Proceedings of the 2013 Conference on Empirical Methods in Natural Language Processing*, pages 1481–1491, Seattle, Washington, USA.

Filippova, K. and Strube, M. (2009). Tree Linearization in English: Improving Language Model Based Approaches. In *Proceedings of the 2009 Conference of the North American*

Chapter of the Association for Computational Linguistics: Human Language Technologies, pages 225–228, Boulder, Colorado, USA.

Finch, A., Akiba, Y., and Sumita, E. (2004). How Does Automatic Machine Translation Evaluation Correlate with Human Scoring as the Number of Reference Translations Increases? In *Proceedings of the 4th International Conference on Language Resources and Evaluation*, Lisbon, Portugal.

Fomicheva, M. and Specia, L. (2016). Reference Bias in Monolingual Machine Translation Evaluation. In *Proceedings of the 54th Annual Meeting of the Association for Computational Linguistics*, pages 77–82, Berlin, Germany. Association for Computational Linguistics.

Fomicheva, M. and Specia, L. (2019). Taking MT Evaluation Metrics to Extremes: Beyond Correlation with Human Judgments. *Computational Linguistics*, 45(3):515–558.

Freitag, M., Grangier, D., and Caswell, I. (2020). BLEU Might Be Guilty but References Are Not Innocent. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 61–71, Online.

Futrell, R., Wilcox, E., Morita, T., Qian, P., Ballesteros, M., and Levy, R. (2019). Neural Language Models as Psycholinguistic Subjects: Representations of Syntactic State. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 32–42, Minneapolis, Minnesota, USA.

G. Snover, M., Madnani, N., Dorr, B., and Schwartz, R. (2009). TER-Plus: Paraphrase, Semantic, and Alignment Enhancements to Translation Edit Rate. *Machine Translation*, 23(2–3):117–127.

Galanis, D. and Androutsopoulos, I. (2007). Generating Multilingual Descriptions from Linguistically Annotated OWL Ontologies: the NaturalOWL System. In *Proceedings of the 11th European Workshop on Natural Language Generation*, pages 143–146, Saarbrücken, Germany.

Gardent, C. and Perez-Beltrachini, L. (2017). A Statistical, Grammar-based Approach to Microplanning. *Computational Linguistics*, 43(1):1–30.

Gardent, C., Shimorina, A., Narayan, S., and Perez-Beltrachini, L. (2017). The WebNLG Challenge: Generating Text from RDF Data. In *Proceedings of the 10th International Conference on Natural Language Generation*, pages 124–133, Santiago de Compostela, Spain.

Gatt, A. and Krahmer, E. (2018). Survey of the State-of-the-art in Natural Language Generation: Core Tasks, Applications and Evaluation. *Journal of Artificial Intelligence Research*, 61(1):65–170.

Gehrmann, S., Dai, F. Z., Elder, H., and Rush, A. M. (2018). End-to-end Content and Plan Selection for Natural Language Generation. In *E2E NLG Challenge System Descriptions*.

Giménez, J. and Màrquez, L. (2007). Linguistic Features for Automatic Evaluation of Heterogenous MT Systems. In *Proceedings of the 2nd Workshop on Statistical Machine Translation*, pages 256–264, Prague, Czech Republic.

Gkatzia, D. and Mahamood, S. (2015). A Snapshot of NLG Evaluation Practices 2005–2014. In *Proceedings of the 15th European Workshop on Natural Language Generation (ENLG)*, pages 57–60, Brighton, UK.

Gneezy, U. and Rustichini, A. (2000). Pay Enough or Don't Pay at All. *The Quarterly Journal of Economics*, 115(3):791–810.

Goldberg, Y. and Elhadad, M. (2010). An Efficient Algorithm for Easy-first Non-directional Dependency Parsing. In *Proceedings of the 2010 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 742–750, Los Angeles, California, USA.

Goodman, J., Cryder, C., and Cheema, A. (2013). Data Collection in a Flat World: the Strengths and Weaknesses of Mechanical Turk Samples. *Journal of Behavioral Decision Making*, 26:213–224.

Graham, Y., Baldwin, T., and Mathur, N. (2015). Accurate Evaluation of Segment-level Machine Translation Metrics. In *Proceedings of the 2015 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 1183–1191, Denver, Colorado, USA.

Green, N. (2006). Generation of Biomedical Arguments for Lay Readers. In *Proceedings of the 4th International Natural Language Generation Conference*, pages 114–121.

Gu, J., Lu, Z., Li, H., and Li, V. O. (2016). Incorporating Copying Mechanism in Sequence-to-sequence Learning. In *Proceedings of the 54th Annual Meeting of the Association for Computational Linguistics*, pages 1631–1640, Berlin, Germany.

Guo, Y., van Genabith, J., and Wang, H. (2008). Dependency-based N-gram Models for General Purpose Sentence Realisation. In *Proceedings of the 22nd International Conference on Computational Linguistics (COLING 2008)*, pages 297–304, Manchester, UK.

Habernal, I., Hannemann, R., Pollak, C., Klamm, C., Pauli, P., and Gurevych, I. (2017). Argotario: Computational Argumentation Meets Serious Games. In *Proceedings of the 2017 Conference on Empirical Methods in Natural Language Processing: System Demonstrations*, pages 7–12, Copenhagen, Denmark.

Hale, J., Dyer, C., Kuncoro, A., and Brennan, J. (2018). Finding Syntax in Human Encephalography with Beam Search. In *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 2727–2736, Melbourne, Australia.

Hamari, J. (2019). *Gamification*, pages 1–3. American Cancer Society.

Hastie, H. and Belz, A. (2014). A Comparative Evaluation Methodology for NLG in Interactive Systems. In *Proceedings of the 9th International Conference on Language Resources and Evaluation*, pages 4004–4011, Reykjavik, Iceland.

He, W., Wang, H., Guo, Y., and Liu, T. (2009). Dependency Based Chinese Sentence Realization. In *Proceedings of the Joint Conference of the 47th Annual Meeting of the ACL and the 4th International Joint Conference on Natural Language Processing of the AFNLP*, pages 809–816, Suntec, Singapore.

Herbrich, R., Minka, T., and Graepel, T. (2006). Trueskill™: a Bayesian Skill Rating System. In *Proceedings of the 19th International Conference on Neural Information Processing Systems*, pages 569–576, Cambridge, MA, USA.

Hermann, K. M., Kočiský, T., Grefenstette, E., Espeholt, L., Kay, W., Suleyman, M., and Blunsom, P. (2015). Teaching Machines to Read and Comprehend. In *Proceedings of the 28th International Conference on Neural Information Processing Systems*, pages 1693–1701, Montréal, Canada.

Hildebrand, M., Brinkerink, M., Gligorov, R., van Steenbergen, M., Huijkman, J., and Oomen, J. (2013). Waisda? Video Labeling Game. In *Proceedings of the 21st ACM International Conference on Multimedia*, pages 823–826, Barcelona, Spain.

Hill, F., Bordes, A., Chopra, S., and Weston, J. (2016). The Goldilocks Principle: Reading Children's Books with Explicit Memory Representations. In *Proceedings of the 4th International Conference on Learning Representations*, San Juan, Puerto Rico.

Hirschmann, F., Nam, J., and Fürnkranz, J. (2016). What Makes Word-level Neural Machine Translation Hard: a Case Study on English-German Translation. In *Proceedings of COLING 2016, the 26th International Conference on Computational Linguistics: Technical Papers*, pages 3199–3208, Osaka, Japan.

Hochreiter, S. and Schmidhuber, J. (1997). Long Short-term Memory. *Neural Computation*, 9(8):1735–1780.

Hou, Y., Liu, Y., Che, W., and Liu, T. (2018). Sequence-to-sequence Data Augmentation for Dialogue Language Understanding. In *Proceedings of the 27th International Conference on Computational Linguistics*, pages 1234–1245, Santa Fe, New Mexico, USA.

Hovy, D., Berg-Kirkpatrick, T., Vaswani, A., and Hovy, E. (2013). Learning Whom to Trust with MACE. In *Proceedings of the 2013 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 1120–1130, Atlanta, Georgia.

Hovy, E. H. (1988). *Generating Natural Language under Pragmatic Constraints*. L. Erlbaum Associates Inc., Hillsdale, NJ, USA.

Hovy, E. H. (1993). Automated Discourse Generation Using Discourse Structure Relations. *Artificial Intelligence*, 63(1):341–385.

Howard, J. and Ruder, S. (2018). Universal Language Model Fine-tuning for Text Classification. In *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics*, pages 328–339, Melbourne, Australia.

Hu, Z., Tan, B., Salakhutdinov, R. R., Mitchell, T. M., and Xing, E. P. (2019). Learning Data Manipulation for Augmentation and Weighting. In *Advances in Neural Information Processing Systems*, volume 32, pages 15764–15775, Montréal, Canada.

Hunter, J., Freer, Y., Gatt, A., Reiter, E., Sripada, S., and Sykes, C. (2012). Automatic Generation of Natural Language Nursing Shift Summaries in Neonatal Intensive Care: BT-Nurse. *Artificial Intelligence in Medicine*, 56(3):157–172.

Inui, K., Tokunaga, T., and Tanaka, H. (1992). *Text Revision: a Model and Its Implementation*, pages 215–230. Springer Berlin Heidelberg.

Ipeirotis, P. G. and Gabrilovich, E. (2014). Quizz: Targeted Crowdsourcing with a Billion (Potential) Users. In *Proceedings of the 23rd International Conference on World Wide Web*, WWW '14, pages 143–154, Seoul, Korea.

Isozaki, H., Hirao, T., Duh, K., Sudoh, K., and Tsukada, H. (2010). Automatic Evaluation of Translation Quality for Distant Language Pairs. In *Proceedings of the 2010 Conference on Empirical Methods in Natural Language Processing*, pages 944–952, Cambridge, MA, USA.

Iyer, S., Konstas, I., Cheung, A., and Zettlemoyer, L. (2016). Summarizing Source Code Using a Neural Attention Model. In *Proceedings of the 54th Annual Meeting of the Association for Computational Linguistics*, pages 2073–2083, Berlin, Germany.

Jiang, J. and Zhai, C. (2007). Instance Weighting for Domain Adaptation in NLP. In *Proceedings of the 45th Annual Meeting of the Association of Computational Linguistics*, pages 264–271, Prague, Czech Republic.

Jones, K. S. and Galliers, J. R. (1995). Evaluating Natural Language Processing Systems: an Analysis and Review. In *Lecture Notes in Artificial Intelligence*. Springer.

Juraska, J., Karagiannis, P., Bowden, K., and Walker, M. (2018). A Deep Ensemble Model with Slot Alignment for Sequence-to-sequence Natural Language Generation. In *Proceedings of the 2018 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 152–162, New Orleans, Louisiana, USA.

Jurgens, D. (2013). Embracing Ambiguity: a Comparison of Annotation Methodologies for Crowdsourcing Word Sense Labels. In *Proceedings of the 2013 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 556–562, Atlanta, Georgia.

Kamigaito, H. and Okumura, M. (2020). Syntactically Look-ahead Attention Network for Sentence Compression. arXiv preprint 2002.01145v2.

Kaplan, R. M. and Kay, M. (1994). Regular Models of Phonological Rule Systems. *Computational Linguistics*, 20(3):331–378.

Khadivi, S. and Ney, H. (2005). Automatic Filtering of Bilingual Corpora for Statistical Machine Translation. In *Natural Language Processing and Information Systems*, pages 263–274, Berlin, Heidelberg.

Khayrallah, H. and Koehn, P. (2018). On the Impact of Various Types of Noise on Neural Machine Translation. In *Proceedings of the 2nd Workshop on Neural Machine Translation and Generation*, pages 74–83, Melbourne, Australia.

Kilickaya, M., Erdem, A., Ikizler-Cinbis, N., and Erdem, E. (2017). Re-evaluating Automatic Metrics for Image Captioning. In *Proceedings of the 15th Conference of the European Chapter of the Association for Computational Linguistics*, pages 199–209, Valencia, Spain.

Kim, Y.-B., Stratos, K., and Kim, D. (2017). Adversarial Adaptation of Synthetic or Stale Data. In *Proceedings of the 55th Annual Meeting of the Association for Computational Linguistics*, pages 1297–1307, Vancouver, Canada.

King, D. and White, M. (2018). The OSU Realizer for SRST '18: Neural Sequence-to-sequence Inflection and Incremental Locality-based Linearization. In *Proceedings of the 1st Workshop on Multilingual Surface Realisation*, pages 39–48, Melbourne, Australia.

Kingma, D. and Ba, J. (2015). Adam: a Method for Stochastic Optimization. In *Proceedings of the 3rd International Conference on Learning Representations*, San Diego, USA.

Kipf, T. N. and Welling, M. (2017). Semi-supervised Classification with Graph Convolutional Networks. In *Proceedings of the 5th International Conference on Learning Representations*, Toulon, France.

Knight, K. and Luk, S. K. (1994). Building a Large-scale Knowledge Base for Machine Translation. In *Proceedings of the 12th AAAI Conference on Artificial Intelligence*, pages 773–778, Seattle, Washington, USA.

Knight, K. and Marcu, D. (2000). Statistics-based Summarization — Step One: Sentence Compression. In *Proceedings of the 17th National Conference on Artificial Intelligence and 12th Conference on Innovative Applications of Artificial Intelligence*, pages 703–710, Austin, Texas, USA.

Koehn, P., Hoang, H., Birch, A., Callison-Burch, C., Federico, M., Bertoldi, N., Cowan, B., Shen, W., Moran, C., Zens, R., Dyer, C., Bojar, O., Constantin, A., and Herbst, E. (2007). Moses: Open Source Toolkit for Statistical Machine Translation. In *Proceedings of the 45th Annual Meeting of the Association for Computational Linguistics*, pages 177–180, Prague, Czech Republic.

Koehn, P. and Knowles, R. (2017). Six Challenges for Neural Machine Translation. In *Proceedings of the 1st Workshop on Neural Machine Translation*, pages 28–39, Vancouver, Canada.

Konstas, I., Iyer, S., Yatskar, M., Choi, Y., and Zettlemoyer, L. (2017). Neural AMR: Sequence-to-sequence Models for Parsing and Generation. In *Proceedings of the 55th Annual Meeting of the Association for Computational Linguistics*, pages 146–157, Vancouver, Canada.

Konstas, I. and Lapata, M. (2012). Unsupervised Concept-to-text Generation with Hypergraphs. In *Proceedings of the 2012 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 752–761, Montréal, Canada.

Konstas, I. and Lapata, M. (2013). Inducing Document Plans for Concept-to-text Generation. In *Proceedings of the 2013 Conference on Empirical Methods in Natural Language Processing*, pages 1503–1514, Seattle, Washington, USA.

Koskenniemi, K. (1983). Two-level Morphology: a General Computational Model for Word-form Recognition and Production. *Publications*, 11:1–160.

Krahmer, E. and van Deemter, K. (2012). Computational Generation of Referring Expressions: a Survey. *Computational Linguistics*, 38(1):173–218.

Kromann, M. T. (2003). The Danish Dependency Treebank and the DTAG Treebank Tool. In *Proceedings of the 2nd Workshop on Treebanks and Linguistic Theories*, pages 217–220, Vaxjo, Sweden.

Kusner, M., Sun, Y., Kolkin, N., and Weinberger, K. (2015). From Word Embeddings to Document Distances. In *Proceedings of the 32nd International Conference on Machine Learning*, volume 37, pages 957–966, Lille, France.

Kuznetsov, S. (2006). Motivations of Contributors to Wikipedia. *ACM SIGCAS*, 36(2):1–7.

Lai, G., Xie, Q., Liu, H., Yang, Y., and Hovy, E. (2017). RACE: Large-scale ReAding Comprehension Dataset from Examinations. In *Proceedings of the 2017 Conference on Empirical Methods in Natural Language Processing*, pages 785–794, Copenhagen, Denmark.

Lampouras, G. and Vlachos, A. (2016). Imitation Learning for Language Generation from Unaligned Data. In *Proceedings of the 26th International Conference on Computational Linguistics: Technical Papers*, pages 1101–1112, Osaka, Japan.

Lan, Z., Chen, M., Goodman, S., Gimpel, K., Sharma, P., and Soricut, R. (2020). ALBERT: A Lite BERT for Self-supervised Learning of Language Representations. In *Proceedings of the 8th International Conference on Learning Representations*, Addis Ababa, Ethiopia.

Langkilde, I. and Knight, K. (1998). Generation That Exploits Corpus-based Statistical Knowledge. In *36th Annual Meeting of the Association for Computational Linguistics and 17th International Conference on Computational Linguistics*, pages 704–710, Montréal, Canada.

Langkilde-Geary, I. (2002). An Empirical Verification of Coverage and Correctness for a General-purpose Sentence Generator. In *Proceedings of the International Natural Language Generation Conference*, pages 17–24, Harriman, New York, USA.

Lapata, M. (2006). Automatic Evaluation of Information Ordering: Kendall's Tau. *Computational Linguistics*, 32(4):471–484.

Leason, T. (2009). Steve: the Art Museum Social Tagging Project: a Report on the Tag Contributor Experience. In *Proceedings of Museums and the Web 2009*, Indianapolis, Indiana, USA.

Lebret, R., Grangier, D., and Auli, M. (2016). Neural Text Generation from Structured Data with Application to the Biography Domain. In *Proceedings of the 2016 Conference on Empirical Methods in Natural Language Processing*, pages 1203–1213, Austin, Texas, USA.

Lemon, O. (2008). Adaptive Natural Language Generation in Dialogue using Reinforcement Learning. In *Proceedings of LONDIAL*, pages 141–148, London, UK.

Lester, J. C. and Porter, B. W. (1997). Developing and Empirically Evaluating Robust Explanation Generators: the Knight Experiments. *Computational Linguistics*, 23(1):65–101.

Levenshtein, V. (1966). Binary Codes Capable of Correcting Deletions, Insertions and Reversals. *Soviet Physics Doklady*, 10(8):707–710. Doklady Akademii Nauk SSSR, V163 No4 845-848 1965.

Levin, P., Dhanuka, N., Khalil, T., Kovalev, F., and Khalilov, M. (2017). Toward a Full-scale Neural Machine Translation in Production: the Booking.com Use Case. arXiv preprint 1709.05820.

Lewis, M., Liu, Y., Goyal, N., Ghazvininejad, M., Mohamed, A., Levy, O., Stoyanov, V., and Zettlemoyer, L. (2020). BART: Denoising Sequence-to-sequence Pretraining for Natural Language Generation, Translation, and Comprehension. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 7871–7880, Online.

Li, J., Chen, X., Hovy, E., and Jurafsky, D. (2016). Visualizing and Understanding Neural Models in NLP. In *Proceedings of the 2016 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 681–691, San Diego, California, USA.

Li, Y., Baldwin, T., and Cohn, T. (2018). Towards Robust and Privacy-preserving Text Representations. In *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics*, pages 25–30, Melbourne, Australia.

Liang, P., Jordan, M., and Klein, D. (2009). Learning Semantic Correspondences with Less Supervision. In *Proceedings of the Joint Conference of the 47th Annual Meeting of the ACL and the 4th International Joint Conference on Natural Language Processing of the AFNLP*, pages 91–99, Suntec, Singapore.

Liddell, T. M. and Kruschke, J. K. (2018). Analyzing Ordinal Data with Metric Models: What Could Possibly Go Wrong? *Journal of Experimental Social Psychology*, 79:328–348.

Lin, C.-Y. and Hovy, E. (2003). Automatic Evaluation of Summaries Using N-gram Co-occurrence Statistics. In *Proceedings of the 2003 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 150–157, Edmonton, Canada.

Lin, C.-Y. and Och, F. J. (2004). ORANGE: a Method for Evaluating Automatic Evaluation Metrics for Machine Translation. In *Proceedings of the 20th International Conference on Computational Linguistics*, pages 501–507, Geneva, Switzerland.

Lin, T.-Y., Maire, M., Belongie, S., Hays, J., Perona, P., Ramanan, D., Dollár, P., and Zitnick, C. L. (2014). Microsoft COCO: Common Objects in Context. In *Proceedings of the 2014 European Conference on Computer Vision*, pages 740–755, Zurich, Switzerland.

Lin, Z., Feng, M., dos Santos, C. N., Yu, M., Xiang, B., Zhou, B., and Bengio, Y. (2017). A Structured Self-attentive Sentence Embedding. In *Proceedings of the 5th International Conference on Learning Representations*, Toulon, France.

Liu, C.-W., Lowe, R., Serban, I., Noseworthy, M., Charlin, L., and Pineau, J. (2016). How NOT to Evaluate Your Dialogue System: an Empirical Study of Unsupervised Evaluation Metrics for Dialogue Response Generation. In *Proceedings of the 2016 Conference on Empirical Methods in Natural Language Processing*, pages 2122–2132, Austin, Texas, USA.

Liu, H., Yin, Q., and Wang, W. Y. (2019a). Towards Explainable NLP: a Generative Explanation Framework for Text Classification. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 5570–5581, Florence, Italy.

Liu, J., Lin, Y., Liu, Z., and Sun, M. (2019b). XQA: a Cross-lingual Open-domain Question Answering Dataset. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 2358–2368, Florence, Italy.

Liu, Y., Ott, M., Goyal, N., Du, J., Joshi, M., Chen, D., Levy, O., Lewis, M., Zettlemoyer, L., and Stoyanov, V. (2019c). RoBERTa: A Robustly Optimized BERT Pretraining Approach. arXiv preprint 1907.11692.

Liu, Y., Zhang, Y., Che, W., and Qin, B. (2015). Transition-based Syntactic Linearization. In *Proceedings of the 2015 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 113–122, Denver, Colorado, USA.

Luong, T., Pham, H., and Manning, C. D. (2015). Effective Approaches to Attention-based Neural Machine Translation. In *Proceedings of the 2015 Conference on Empirical Methods in Natural Language Processing*, pages 1412–1421, Lisbon, Portugal.

Luong, T., Socher, R., and Manning, C. (2013). Better Word Representations with Recursive Neural Networks for Morphology. In *Proceedings of the 17th Conference on Computational Natural Language Learning*, pages 104–113, Sofia, Bulgaria.

Ma, Q., Wei, J., Bojar, O., and Graham, Y. (2019a). Results of the WMT19 Metrics Shared Task: Segment-level and Strong MT Systems Pose Big Challenges. In *Proceedings of the 4th Conference on Machine Translation*, pages 62–90, Florence, Italy.

Ma, S., Yang, P., Liu, T., Li, P., Zhou, J., and Sun, X. (2019b). Key Fact as Pivot: a Two-stage Model for Low Resource Table-to-text Generation. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 2047–2057, Florence, Italy.

Maas, A. L., Hannun, A. Y., and Ng, A. Y. (2013). Rectifier Nonlinearities Improve Neural Network Acoustic Models. In *ICML Workshop on Deep Learning for Audio, Speech and Language Processing*, Atlanta, USA.

Mahamood, S. and Reiter, E. (2011). Generating Affective Natural Language for Parents of Neonatal Infants. In *Proceedings of the 13th European Workshop on Natural Language Generation*, pages 12–21, Nancy, France.

Mairesse, F. and Young, S. (2014). Stochastic Language Generation in Dialogue Using Factored Language Models. *Computational Linguistics*, 40(4):763–799.

Mani, I., Gates, B., and Bloedorn, E. (1999). Improving Summaries by Revising Them. In *Proceedings of the 37th Annual Meeting of the Association for Computational Linguistics on Computational Linguistics*, pages 558–565, College Park, Maryland, USA.

Mann, W. C. and Thompson, S. A. (1988). Rhetorical Structure Theory: Toward a Functional Theory of Text Organization. *Text — Interdisciplinary Journal for the Study of Discourse*, 8(3):243–281.

Manning, C. D. (2011). Part-of-speech Tagging from 97 Percent to 100 Percent: Is It Time for Some Linguistics? In *Proceedings of the 12th International Conference on Computational Linguistics and Intelligent Text Processing*, pages 171–189, Tokyo, Japan.

Marcheggiani, D. and Perez-Beltrachini, L. (2018). Deep Graph Convolutional Encoders for Structured Data to Text Generation. In *Proceedings of the 11th International Conference on Natural Language Generation*, pages 1–9, Tilburg University, Netherlands.

Marciniak, T. and Strube, M. (2005). Beyond the Pipeline: Discrete Optimization in NLP. In *Proceedings of the 9th Conference on Computational Natural Language Learning*, pages 136–143, Ann Arbor, Michigan, USA.

Marr, D. and Brenner, S. (1976). Early Processing of Visual Information. *Philosophical Transactions of the Royal Society of London. B, Biological Sciences*, 275(942):483–519.

Mason, W. and Suri, S. (2012). Conducting Behavioral Research on Amazon's Mechanical Turk. *Behavior Research Methods*, 44(1):1–23.

Mason, W. and Watts, D. J. (2009). Financial Incentives and the "Performance of Crowds". In *Proceedings of the ACM SIGKDD Workshop on Human Computation*, pages 77–85, Paris, France.

Mathur, N., Baldwin, T., and Cohn, T. (2020). Tangled up in BLEU: Reevaluating the Evaluation of Automatic Machine Translation Evaluation Metrics. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 4984–4997, Online.

May, J. and Priyadarshi, J. (2017). SemEval-2017 Task 9: Abstract Meaning Representation Parsing and Generation. In *Proceedings of the 11th International Workshop on Semantic Evaluation (SemEval-2017)*, pages 536–545, Vancouver, Canada.

McDonald, D. D. (1983). Description Directed Control: Its Implications for Natural Language Generation. *Computers and Mathematics with Applications*, 9(1):111–129.

McDonald, R. (2006). Discriminative Sentence Compression with Soft Syntactic Evidence. In *Proceedings of the 11th Conference of the European Chapter of the Association for Computational Linguistics*, Trento, Italy.

McKeown, K. (1985). *Text Generation*. Cambridge University Press, Cambridge, UK.

Mei, H., Bansal, M., and Walter, M. R. (2016). What to Talk about and How? Selective Generation using LSTMs with Coarse-to-fine Alignment. In *Proceedings of the 2016 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 720–730, San Diego, California, USA.

Meteer, M. W. (1991). Bridging the Generation Gap Between Text Planning and Linguistic Realization. *Computational Intelligence*, 7(4):296–304.

Meyers, A., Reeves, R., Macleod, C., Szekely, R., Zielinska, V., Young, B., and Grishman, R. (2004). The NomBank Project: an Interim Report. In *Proceedings of the Workshop Frontiers in Corpus Annotation at HLT-NAACL 2004*, pages 24–31, Boston, Massachusetts, USA.

Mikolov, T., Sutskever, I., Chen, K., Corrado, G. S., and Dean, J. (2013). Distributed Representations of Words and Phrases and Their Compositionality. In *Advances in Neural Information Processing Systems*, volume 26, pages 3111–3119, Montréal, Canada.

Mille, S., Belz, A., Bohnet, B., Graham, Y., Pitler, E., and Wanner, L. (2018). The 1st Multilingual Surface Realisation Shared Task (SR'18): Overview and Evaluation Results. In *Proceedings of the 1st Workshop on Multilingual Surface Realisation*, pages 1–12, Melbourne, Australia.

Mille, S., Belz, A., Bohnet, B., Graham, Y., and Wanner, L. (2019). The 2nd Multilingual Surface Realisation Shared Task (SR'19): Overview and Evaluation Results. In *Proceedings of the 2nd Workshop on Multilingual Surface Realisation*, pages 1–17, Hong Kong, China.

Mnih, V., Heess, N., Graves, A., and kavukcuoglu, k. (2014). Recurrent Models of Visual Attention. In *Advances in Neural Information Processing Systems*, volume 27, Montréal, Canada.

Moore, J. D. and Paris, C. L. (1993). Planning Text for Advisory Dialogues: Capturing Intentional and Rhetorical Information. *Computational Linguistics*, 19(4):651–694.

Morschheuser, B., Hamari, J., and Koivisto, J. (2016). Gamification in Crowdsourcing: a Review. In *Proceedings of the 49th Hawaii International Conference on System Sciences*, pages 4375–4384, Koloa, HI, USA.

Mrini, K., Dernoncourt, F., Tran, Q. H., Bui, T., Chang, W., and Nakashole, N. (2020). Rethinking Self-attention: Towards Interpretability in Neural Parsing. In *Findings of the Association for Computational Linguistics: EMNLP 2020*, pages 731–742, Online.

Nair, V. and Hinton, G. E. (2010). Rectified Linear Units Improve Restricted Boltzmann Machines. In *Proceedings of the 27th International Conference on Machine Learning*, pages 807–814, Haifa, Israel.

Nallapati, R., Zhou, B., dos Santos, C., Gulçehre, Ç., and Xiang, B. (2016). Abstractive Text Summarization Using Sequence-to-sequence RNNs and Beyond. In *Proceedings of The 20th SIGNLL Conference on Computational Natural Language Learning*, pages 280–290, Berlin, Germany.

Napoles, C., Van Durme, B., and Callison-Burch, C. (2011). Evaluating Sentence Compression: Pitfalls and Suggested Remedies. In *Proceedings of the Workshop on Monolingual Text-To-Text Generation*, pages 91–97, Portland, Oregon.

Narayan, S. and Gardent, C. (2020). Deep Learning Approaches to Text Production. *Synthesis Lectures on Human Language Technologies*, 13(1):1–199.

Nenkova, A., Chae, J., Louis, A., and Pitler, E. (2010). *Structural Features for Predicting the Linguistic Quality of Text*, pages 222–241. Springer Berlin Heidelberg.

Nie, F., Yao, J.-G., Wang, J., Pan, R., and Lin, C.-Y. (2019). A Simple Recipe Towards Reducing Hallucination in Neural Surface Realisation. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 2673–2679, Florence, Italy.

Nisioi, S., Štajner, S., Ponzetto, S. P., and Dinu, L. P. (2017). Exploring Neural Text Simplification Models. In *Proceedings of the 55th Annual Meeting of the Association for Computational Linguistics*, pages 85–91, Vancouver, Canada.

Nivre, J. (2003). An Efficient Algorithm for Projective Dependency Parsing. In *Proceedings of the 8th International Conference on Parsing Technologies*, pages 149–160, Nancy, France.

Nivre, J. (2006). Constraints on Non-projective Dependency Parsing. In *Proceedings of the 11th Conference of the European Chapter of the Association for Computational Linguistics*, pages 73–80, Trento, Italy.

Nov, O. (2007). What Motivates Wikipedians? *ACM Computing Surveys*, 50(11):60–64.

Novikova, J., Dušek, O., and Rieser, V. (2017a). The E2E Dataset: New Challenges for End-to-end Generation. In *Proceedings of the 18th Annual SIGdial Meeting on Discourse and Dialogue*, pages 201–206, Saarbrücken, Germany.

Novikova, J., Dušek, O., and Rieser, V. (2018). RankME: Reliable Human Ratings for Natural Language Generation. In *Proceedings of the 2018 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 72–78, New Orleans, Louisiana, USA.

Novikova, J., Dušek, O., Cercas Curry, A., and Rieser, V. (2017b). Why We Need New Evaluation Metrics for NLG. In *Proceedings of the 2017 Conference on Empirical Methods in Natural Language Processing*, pages 2241–2252, Copenhagen, Denmark.

Novikova, J., Lemon, O., and Rieser, V. (2016). Crowd-sourcing NLG Data: Pictures Elicit Better Data. In *Proceedings of the 9th International Natural Language Generation Conference*, pages 265–273, Edinburgh, UK.

Oraby, S., Reed, L., Tandon, S., T.S., S., Lukin, S., and Walker, M. (2018). TNT-NLG, System 1: Using a Statistical NLG to Massively Augment Crowd-sourced Data for Neural Generation. In *E2E NLG Challenge System Descriptions*.

Palmer, M., Gildea, D., and Kingsbury, P. (2005). The Proposition Bank: an Annotated Corpus of Semantic Roles. *Computational Linguistics*, 31(1):71–106.

Pan, S. J. and Yang, Q. (2010). A Survey on Transfer Learning. *IEEE Transactions on Knowledge and Data Engineering*, 22(10):1345–1359.

Papineni, K., Roukos, S., Ward, T., and Zhu, W.-J. (2002). BLEU: a Method for Automatic Evaluation of Machine Translation. In *Proceedings of 40th Annual Meeting of the Association for Computational Linguistics*, pages 311–318, Philadelphia, Pennsylvania, USA.

Parikh, A., Täckström, O., Das, D., and Uszkoreit, J. (2016). A Decomposable Attention Model for Natural Language Inference. In *Proceedings of the 2016 Conference on Empirical Methods in Natural Language Processing*, pages 2249–2255, Austin, Texas.

Paszke, A., Gross, S., Massa, F., Lerer, A., Bradbury, J., Chanan, G., Killeen, T., Lin, Z., Gimelshein, N., Antiga, L., Desmaison, A., Kopf, A., Yang, E., DeVito, Z., Raison, M., Tejani, A., Chilamkurthy, S., Steiner, B., Fang, L., Bai, J., and Chintala, S. (2019). PyTorch: an Imperative Style, High-performance Deep Learning Library. In *Advances in Neural Information Processing Systems*, volume 32, pages 8024–8035, Montréal, Canada.

Pennington, J., Socher, R., and Manning, C. (2014). GloVe: Global Vectors for Word Representation. In *Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 1532–1543, Doha, Qatar.

Peters, M., Neumann, M., Iyyer, M., Gardner, M., Clark, C., Lee, K., and Zettlemoyer, L. (2018). Deep Contextualized Word Representations. In *Proceedings of the 2018 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 2227–2237, New Orleans, Louisiana, USA.

Peyrard, M. (2019). A Simple Theoretical Model of Importance for Summarization. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 1059–1073, Florence, Italy.

Popescu-Belis, A. (2007). Evaluation of NLG: Some Analogies and Differences with MT and Reference Resolution. In *Proceedings of MT Summit XI Workshop on Using Corpora for NLG and MT*, Copenhagen, Danemark.

Portet, F., Reiter, E., Gatt, A., Hunter, J., Sripada, S., Freer, Y., and Sykes, C. (2009). Automatic Generation of Textual Summaries from Neonatal Intensive Care Data. *Artificial Intelligence*, 173(7):789–816.

Pratt, L. (1993). Discriminability-based Transfer Between Neural Networks. In *Advances in Neural Information Processing Systems*, volume 5, pages 204–211, Montréal, Canada.

Puduppully, R., Zhang, Y., and Shrivastava, M. (2016). Transition-based Syntactic Linearization with Lookahead Features. In *Proceedings of the 2016 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 488–493, San Diego, California, USA.

Puzikov, Y., Gardent, C., Dagan, I., and Gurevych, I. (2019). Revisiting the Binary Linearization Technique for Surface Realization. In *Proceedings of the 12th International Conference on Natural Language Generation*, pages 268–278, Tokyo, Japan.

Puzikov, Y. and Gurevych, I. (2018a). BinLin: a Simple Method of Dependency Tree Linearization. In *Proceedings of the 1st Workshop on Multilingual Surface Realisation*, pages 13–28, Melbourne, Australia.

Puzikov, Y. and Gurevych, I. (2018b). E2E NLG Challenge: Neural Models vs. Templates. In *Proceedings of the 11th International Conference on Natural Language Generation*, pages 463–471, Tilburg University, Netherlands.

Qin, Y. and Specia, L. (2015). Truly Exploring Multiple References for Machine Translation Evaluation. In *Proceedings of the 18th Annual Conference of the European Association for Machine Translation*, pages 113–120, Antalya, Turkey.

Radford, A., Narasimhan, K., Salimans, T., and Sutskever, I. (2018). Improving Language Understanding with Unsupervised Learning.

Radford, A., Wu, J., Child, R., Luan, D., Amodei, D., and Sutskever, I. (2019). Language Models Are Unsupervised Multitask Learners.

Raffel, C., Shazeer, N., Roberts, A., Lee, K., Narang, S., Matena, M., Zhou, Y., Li, W., and Liu, P. J. (2020). Exploring the Limits of Transfer Learning with a Unified Text-to-text Transformer. *Journal of Machine Learning Research*, 21(140):1–67.

Rajpurkar, P., Zhang, J., Lopyrev, K., and Liang, P. (2016). SQuAD: 100,000+ Questions for Machine Comprehension of Text. In *Proceedings of the 2016 Conference on Empirical Methods in Natural Language Processing*, pages 2383–2392, Austin, Texas.

Ratnaparkhi, A. (2000). Trainable Methods for Surface Natural Language Generation. In *Proceedings of the 2000 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 194–201, Seattle, Washington, USA.

Reimers, N. and Gurevych, I. (2017). Reporting Score Distributions Makes a Difference: Performance Study of LSTM-networks for Sequence Tagging. In *Proceedings of the 2017 Conference on Empirical Methods in Natural Language Processing*, pages 338–348, Copenhagen, Denmark.

Reiter, E. (1994). Has a Consensus NL Generation Architecture Appeared, and is It Psycholinguistically Plausible? In *Proceedings of the 7th International Workshop on Natural Language Generation*, pages 163–170, Kennebunkport, Maine.

Reiter, E. and Belz, A. (2009). An Investigation into the Validity of Some Metrics for Automatically Evaluating Natural Language Generation Systems. *Computational Linguistics*, 35(4):529–558.

Reiter, E. and Dale, R. (2000). *Building Natural Language Generation Systems*. Cambridge University Press, Cambridge, UK.

Reiter, E. and Mellish, C. (1993). Optimising the Costs and Benefits of Natural Language Generation. In *Proceedings of the 13th International Joint Conference on Artificial Intelligence*, pages 1164–1169, Chambéry, France.

Reiter, E., Mellish, C., and Levine, J. (1995). Automatic Generation of Technical Documentation. *Applied Artificial Intelligence*, 9(3):259–287.

Reiter, E., Robertson, R., and Osman, L. M. (2003). Lessons from a Failure: Generating Tailored Smoking Cessation Letters. *Artificial Intelligence*, 144(1):41–58.

Reiter, E. and Sripada, S. (2002). Should Corpora Texts Be Gold Standards for NLG? In *Proceedings of the 2nd International Conference on Natural Language Generation*, pages 97–104, Harriman, New York, USA.

Reiter, E., Sripada, S., Hunter, J., Yu, J., and Davy, I. (2005). Choosing Words in Computer-generated Weather Forecasts. *Artificial Intelligence*, 167(1):137–169. Connecting Language to the World.

Reiter, E. and Thomson, C. (2020). Shared Task on Evaluating Accuracy. In *Proceedings of the 13th International Conference on Natural Language Generation*, pages 227–231, Dublin, Ireland.

Rieser, V. and Lemon, O. (2009). Natural Language Generation as Planning under Uncertainty for Spoken Dialogue Systems. In *Proceedings of the 12th Conference of the European Chapter of the ACL*, pages 683–691, Athens, Greece.

Rieser, V. and Lemon, O. (2011). *Reinforcement Learning for Adaptive Dialogue Systems: A Data-driven Methodology for Dialogue Management and Natural Language Generation*. Springer Publishing Company, Incorporated.

Rumelhart, D. E., Hinton, G. E., and Williams, R. J. (1986). Learning Representations by Back-propagating Errors. *Nature*, 323(6088):533–536.

Rus, V., Wyse, B., Piwek, P., Lintean, M., Stoyanchev, S., and Moldovan, C. (2011). Question Generation Shared Task and Evaluation Challenge — Status Report. In *Proceedings of the 13th European Workshop on Natural Language Generation*, pages 318–320, Nancy, France.

Rush, A. M., Chopra, S., and Weston, J. (2015). A Neural Attention Model for Abstractive Sentence Summarization. In *Proceedings of the 2015 Conference on Empirical Methods in Natural Language Processing*, pages 379–389, Lisbon, Portugal.

Sakaguchi, K., Post, M., and Van Durme, B. (2014). Efficient Elicitation of Annotations for Human Evaluation of Machine Translation. In *Proceedings of the 9th Workshop on Statistical Machine Translation*, pages 1–11, Baltimore, Maryland, USA.

Samek, W., Wiegand, T., and Müller, K.-R. (2018). Explainable Artificial Intelligence: Understanding, Visualizing and Interpreting Deep Learning Models. *ITU Journal: ICT Discoveries*, 1(1):39–48.

Sanh, V., Debut, L., Chaumond, J., and Wolf, T. (2019). DistilBERT, a Distilled Version of BERT: Smaller, Faster, Cheaper and Lighter. arXiv preprint 1910.01108.

Sartorio, F., Satta, G., and Nivre, J. (2013). A Transition-based Dependency Parser Using a Dynamic Parsing Strategy. In *Proceedings of the 51st Annual Meeting of the Association for Computational Linguistics*, pages 135–144, Sofia, Bulgaria.

Schuster, M. and Paliwal, K. (1997). Bidirectional Recurrent Neural Networks. *IEEE Transactions on Signal Processing*, 45(11):2673–2681.

Scott, D. and Moore, J. (2007). An NLG Evaluation Competition? Eight Reasons to Be Cautious. In *Proceedings of the NSF Workshop on Shared Tasks and Comparative Evaluation in Natural Language Generation*, pages 22–23, Arlington, VA, USA.

See, A., Liu, P. J., and Manning, C. D. (2017). Get to the Point: Summarization with Pointer-generator Networks. In *Proceedings of the 55th Annual Meeting of the Association for Computational Linguistics*, pages 1073–1083, Vancouver, Canada.

Sellam, T., Das, D., and Parikh, A. (2020). BLEURT: Learning Robust Metrics for Text Generation. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 7881–7892, Online.

Shah, D. S., Schwartz, H. A., and Hovy, D. (2020). Predictive Biases in Natural Language Processing Models: a Conceptual Framework and Overview. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 5248–5264, Online.

Shorten, C. and Khoshgoftaar, T. M. (2019). A Survey on Image Data Augmentation for Deep Learning. *Journal of Big Data*, 6(1):1–48.

Siddharthan, A., Green, M., van Deemter, K., Mellish, C., and van der Wal, R. (2012). Blogging Birds: Generating Narratives about Reintroduced Species to Promote Public Engagement. In *Proceedings of the 7th International Natural Language Generation Conference*, pages 120–124, Utica, Illinois, USA.

Simpson, E., Do Dinh, E.-L., Miller, T., and Gurevych, I. (2019). Predicting Humorousness and Metaphor Novelty with Gaussian Process Preference Learning. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 5716–5728, Florence, Italy.

Simpson, E. and Gurevych, I. (2019). A Bayesian Approach for Sequence Tagging with Crowds. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing*, pages 1093–1104, Hong Kong, China.

Snover, M., Dorr, B., Schwartz, R., Micciulla, L., and Makhoul, J. (2006). A Study of Translation Edit Rate with Targeted Human Annotation. In *Proceedings of Association for Machine Translation in the Americas*, pages 223–231.

Song, L., Zhang, Y., Wang, Z., and Gildea, D. (2018). A Graph-to-sequence Model for AMR-to-text Generation. In *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics*, pages 1616–1626, Melbourne, Australia.

Sripada, S., Reiter, E., Hunter, J., and Yu, J. (2003a). Exploiting a Parallel Text-data Corpus. In *Proceedings of Corpus Linguistics 2003*, pages 734–743, Lancaster, UK.

Sripada, S. G., Reiter, E., and Davy, I. (2003b). SUMTIME-MOUSAM: Configurable Marine Weather Forecast Generator. *Expert Update*, pages 4–10.

Sripada, S. G., Reiter, E., Davy, I., and Nilssen, K. (2004). Lessons from Deploying NLG Technology for Marine Weather Forecast Text Generation. In *Proceedings of the*

16th European Conference on Artificial Intelligence: Prestigious Applicants of Intelligent Systems, pages 760–764, Amsterdam, Netherlands.

Srivastava, N., Hinton, G., Krizhevsky, A., Sutskever, I., and Salakhutdinov, R. (2014). Dropout: a Simple Way to Prevent Neural Networks from Overfitting. *Journal of Machine Learning Research*, 15(56):1929–1958.

Štajner, S. and Saggion, H. (2013). Readability Indices for Automatic Evaluation of Text Simplification Systems: a Feasibility Study for Spanish. In *Proceedings of the 6th International Joint Conference on Natural Language Processing*, pages 374–382, Nagoya, Japan.

Stanojević, M. and Sima'an, K. (2014). BEER: BEtter Evaluation as Ranking. In *Proceedings of the Ninth Workshop on Statistical Machine Translation*, pages 414–419, Baltimore, Maryland, USA.

Sun, H., Dhingra, B., Zaheer, M., Mazaitis, K., Salakhutdinov, R., and Cohen, W. (2018). Open Domain Question Answering Using Early Fusion of Knowledge Bases and Text. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing*, pages 4231–4242, Brussels, Belgium.

Sun, S., Shapira, O., Dagan, I., and Nenkova, A. (2019). How to Compare Summarizers without Target Length? Pitfalls, Solutions and Re-examination of the Neural Summarization Literature. In *Proceedings of the Workshop on Methods for Optimizing and Evaluating Neural Language Generation*, pages 21–29, Minneapolis, Minnesota, USA.

Sun, Y. (2010). Mining the Correlation between Human and Automatic Evaluation at Sentence Level. In *Proceedings of the 7th conference on International Language Resources and Evaluation*, pages 1726–1730, Valletta, Malta.

Surdeanu, M., Johansson, R., Meyers, A., Màrquez, L., and Nivre, J. (2008). The CoNLL 2008 Shared Task on Joint Parsing of Syntactic and Semantic Dependencies. In *Proceedings of the 12th Conference on Computational Natural Language Learning*, pages 159–177, Manchester, UK.

Sutskever, I., Vinyals, O., and Le, Q. V. (2014). Sequence-to-sequence Learning with Neural Networks. In *Advances in Neural Information Processing Systems*, volume 27, pages 3104–3112, Montréal, Canada.

Tandon, S., T.S., S., Oraby, S., Reed, L., Lukin, S., and Walker, M. (2018). TNT-NLG, System 2: Data Repetition and Meaning Representation Manipulation to Improve Neural Generation. In *E2E NLG Challenge System Descriptions*.

Tang, D., Duan, N., Qin, T., and Zhou, M. (2017). Question Answering and Question Generation as Dual Tasks. arXiv preprint 1706.02027.

Tenney, I., Das, D., and Pavlick, E. (2019). BERT rediscovers the classical NLP pipeline. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 4593–4601, Florence, Italy.

Thompson, H. (1977). Strategy and Tactics: a Model for Language Production. In *Papers from the 13th Regional Meeting of the Chicago Linguistics Society*, volume 13, pages 651–668, Chicago, Illinois, USA.

Tu, Z., Lu, Z., Liu, Y., Liu, X., and Li, H. (2016). Modeling Coverage for Neural Machine Translation. In *Proceedings of the 54th Annual Meeting of the Association for Computational Linguistics*, pages 76–85, Berlin, Germany.

Turian, J. P., Shen, L., and Melamed, I. D. (2003). Evaluation of Machine Translation and Its Evaluation. In *Proceedings of the MT Summit IX*, pages 386–393, New Orleans, Louisiana, USA.

Vajjala, S. and Meurers, D. (2012). On Improving the Accuracy of Readability Classification Using Insights from Second Language Acquisition. In *Proceedings of the 7th Workshop on Building Educational Applications Using NLP*, pages 163–173, Montréal, Canada.

Van der Lee, C., Gatt, A., van Miltenburg, E., Wubben, S., and Krahmer, E. (2019). Best Practices for the Human Evaluation of Automatically Generated Text. In *Proceedings of the 12th International Conference on Natural Language Generation*, pages 355–368, Tokyo, Japan.

Vannella, D., Jurgens, D., Scarfini, D., Toscani, D., and Navigli, R. (2014). Validating and Extending Semantic Knowledge Bases Using Video Games with a Purpose. In *Proceedings of the 52nd Annual Meeting of the Association for Computational Linguistics*, pages 1294–1304, Baltimore, Maryland, USA.

Vaswani, A., Shazeer, N., Parmar, N., Uszkoreit, J., Jones, L., Gomez, A. N., Kaiser, u., and Polosukhin, I. (2017). Attention Is All You Need. In *Proceedings of the 31st International Conference on Neural Information Processing Systems*, pages 6000–6010, Long Beach, California, USA.

Vedantam, R., Zitnick, C. L., and Parikh, D. (2015). CIDEr: Consensus-based Image Description Evaluation. In *Proceedings of the 2015 IEEE Conference on Computer Vision and Pattern Recognition*, pages 4566–4575.

Venhuizen, N. J., Basile, V., Evang, K., and Bos, J. (2013). Gamification for Word Sense Labeling. In *Proceedings of the 10th International Conference on Computational Semantics*, pages 397–403, Potsdam, Germany.

Venturi, G., Bellandi, T., Dell'Orletta, F., and Montemagni, S. (2015). NLP-based Readability Assessment of Health-related Texts: a Case Study on Italian Informed Consent Forms. In *Proceedings of the 6th International Workshop on Health Text Mining and Information Analysis*, pages 131–141, Lisbon, Portugal.

Vinyals, O., Kaiser, L. u., Koo, T., Petrov, S., Sutskever, I., and Hinton, G. (2015). Grammar as a Foreign Language. In *Advances in Neural Information Processing Systems*, volume 28, pages 2773–2781, Montréal, Canada.

Vu, T., Hu, B., Munkhdalai, T., and Yu, H. (2018). Sentence Simplification with Memory-augmented Neural Networks. In *Proceedings of the 2018 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 79–85, New Orleans, Louisiana, USA.

Wang, A., Singh, A., Michael, J., Hill, F., Levy, O., and Bowman, S. (2018). GLUE: a Multi-task Benchmark and Analysis Platform for Natural Language Understanding. In *Proceedings of the 2018 EMNLP Workshop BlackboxNLP: Analyzing and Interpreting Neural Networks for NLP*, pages 353–355, Brussels, Belgium.

Wang, L., Jiang, J., Chieu, H. L., Ong, C. H., Song, D., and Liao, L. (2017). Can Syntax Help? Improving an LSTM-based Sentence Compression Model for New Domains. In *Proceedings of the 55th Annual Meeting of the Association for Computational Linguistics*, pages 1385–1393, Vancouver, Canada.

Wei, J. and Zou, K. (2019). EDA: Easy Data Augmentation Techniques for Boosting Performance on Text Classification Tasks. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing*, pages 6382–6388, Hong Kong, China.

Wen, T.-H., Gasic, M., Mrkšić, N., Su, P.-H., Vandyke, D., and Young, S. (2015). Semantically Conditioned LSTM-based Natural Language Generation for Spoken Dialogue Systems. In *Proceedings of the 2015 Conference on Empirical Methods in Natural Language Processing*, pages 1711–1721, Lisbon, Portugal.

White, M. and Rajkumar, R. (2009). Perceptron Reranking for CCG Realization. In *Proceedings of the 2009 Conference on Empirical Methods in Natural Language Processing*, pages 410–419, Singapore.

White, M., Rajkumar, R., and Martin, S. (2007). Towards Broad Coverage Surface Realization with CCG. In *Proceedings of the 2007 Workshop on Using Corpora for NLG: Language Generation and Machine Translation*, pages 22–30, Copenhagen, Danmark.

Wicentowski, R. (2004). Multilingual Noise-robust Supervised Morphological Analysis Using the Wordframe Model. In *Proceedings of the 7th Meeting of the ACL Special Interest Group in Computational Phonology: Current Themes in Computational Phonology and Morphology*, pages 70–77, Barcelona, Spain.

Williams, R. J. (1992). Simple Statistical Gradient-following Algorithms for Connectionist Reinforcement Learning. *Machine Learning*, 8(3-4):229–256.

Wiseman, S., Shieber, S., and Rush, A. (2017). Challenges in Data-to-document Generation. In *Proceedings of the 2017 Conference on Empirical Methods in Natural Language Processing*, pages 2253–2263, Copenhagen, Denmark.

Wities, R., Shwartz, V., Stanovsky, G., Adler, M., Shapira, O., Upadhyay, S., Roth, D., Martínez-Cámara, E., Gurevych, I., and Dagan, I. (2017). A Consolidated Open Knowledge Representation for Multiple Texts. In *Proceedings of the 2nd Workshop on Linking Models of Lexical, Sentential and Discourse-level Semantics*, pages 12–24, Valencia, Spain.

Wolf, T., Debut, L., Sanh, V., Chaumond, J., Delangue, C., Moi, A., Cistac, P., Rault, T., Louf, R., Funtowicz, M., Davison, J., Shleifer, S., von Platen, P., Ma, C., Jernite, Y., Plu, J., Xu, C., Le Scao, T., Gugger, S., Drame, M., Lhoest, Q., and Rush, A. (2020). Transformers: State-of-the-art Natural Language Processing. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing: System Demonstrations*, pages 38–45, Online.

Wong, Y. W. and Mooney, R. (2007). Generation by Inverting a Semantic Parser that Uses Statistical Machine Translation. In *Proceedings of the 2007 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 172–179, Rochester, New York.

Wubben, S., van den Bosch, A., and Krahmer, E. (2012). Sentence Simplification by Monolingual Machine Translation. In *Proceedings of the 50th Annual Meeting of the Association for Computational Linguistics*, pages 1015–1024, Jeju Island, Korea.

Xu, K., Ba, J. L., Kiros, R., Cho, K., Courville, A., Salakhutdinov, R., Zemel, R. S., and Bengio, Y. (2015a). Show, Attend and Tell: Neural Image Caption Generation with Visual Attention. In *Proceedings of the 32nd International Conference on International Conference on Machine Learning*, pages 2048–2057, Lille, France.

Xu, K., Ba, J. L., Kiros, R., Cho, K., Courville, A., Salakhutdinov, R., Zemel, R. S., and Bengio, Y. (2015b). Show, Attend and Tell: Neural Image Caption Generation with Visual Attention. In *Proceedings of the 32nd International Conference on International Conference on Machine Learning*, pages 2048–2057, Lille, France.

Yamada, H. and Matsumoto, Y. (2003). Statistical Dependency Analysis with Support Vector Machines. In *Proceedings of the 8th International Conference on Parsing Technologies*, pages 195–206, Nancy, France.

Yang, Z., Dai, Z., Yang, Y., Carbonell, J., Salakhutdinov, R. R., and Le, Q. V. (2019). XLNet: Generalized Autoregressive Pretraining for Language Understanding. In *Advances in Neural Information Processing Systems*, volume 32, pages 5754–5764, Vancouver, Canada.

Yannakakis, G. N. and Hallam, J. (2011). Ranking vs. Preference: a Comparative Study of Self-reporting. In *Proceedings of the 2011 International Conference on Affective Computing and Intelligent Interaction*, pages 437–446, Memphis, TN, USA.

Yarowsky, D. and Wicentowski, R. (2000). Minimally Supervised Morphological Analysis by Multimodal Alignment. In *Proceedings of the 38th Annual Meeting on Association for Computational Linguistics*, pages 207–216, Hong Kong.

Yeh, C.-L. and Mellish, C. (1997). An Empirical Study on the Generation of Anaphora in Chinese. *Computational Linguistics*, 23(1):169–190.

Young, R. M. (1999). Using Grice's Maxim of Quantity to Select the Content of Plan Descriptions. *Artificial Intelligence*, 115:215–256.

Yu, Z., Zang, H., and Wan, X. (2020). Homophonic Pun Generation with Lexically Constrained Rewriting. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 2870–2876, Online.

Zhang, T., Kishore, V., Wu, F., Weinberger, K. Q., and Artzi, Y. (2020a). BERTscore: Evaluating text generation with BERT. In *Proceedings of the 8th International Conference on Learning Representations*, Addis Ababa, Ethiopia.

Zhang, X. and Lapata, M. (2014). Chinese Poetry Generation with Recurrent Neural Networks. In *Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing*, pages 670–680, Doha, Qatar.

Zhang, Y. (2013). Partial-tree Linearization: Generalized Word Ordering for Text Synthesis. In *Proceedings of the 23rd International Joint Conference on Artificial Intelligence*, pages 2232–2238, Beijing, China.

Zhang, Y. and Clark, S. (2011). Syntactic Processing Using the Generalized Perceptron and Beam Search. *Computational Linguistics*, 37(1):105–151.

Zhang, Y., Sun, S., Galley, M., Chen, Y.-C., Brockett, C., Gao, X., Gao, J., Liu, J., and Dolan, B. (2020b). DIALOGPT : Large-scale Generative Pretraining For Conversational Response Generation. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics: System Demonstrations*, pages 270–278, Online.

Zhao, Y., Luo, Z., and Aizawa, A. (2018). A Language Model based Evaluator for Sentence Compression. In *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics*, pages 170–175, Melbourne, Australia.

Zhou, Q., Yang, N., Wei, F., Huang, S., Zhou, M., and Zhao, T. (2018). Neural Document Summarization by Jointly Learning to Score and Select Sentences. In *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics*, pages 654–663, Melbourne, Australia.

Zhu, H., Dong, L., Wei, F., Wang, W., Qin, B., and Liu, T. (2019). Learning to Ask Unanswerable Questions for Machine Reading Comprehension. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 4238–4248, Florence, Italy.

Notes on handling research data

According to the "Guidelines on the handling of research data" of the Deutsche Forschungsgemeinschaft[¶], all the data and software related to this dissertation are archived and made pubicly available where possible.

The following research data has been made freely available:

- **Software**
  - The software required for the experiments described in Section 4.3 and Section 4.4 is available under the Apache License 2.0 license at `https://github.com/UKPLab/e2e-nlg-challenge-2017`.
  - The software required for the experiments described in Section 6.4 is available under the Apache License 2.0 license at `https://github.com/UKPLab/inlg2019-revisiting-binlin` and `https://github.com/UKPLab/acl2018-msr-workshop-binlin`.
  - The software required for the experiments described in Chapter 5 cannot be made available due to Intellectual Property restrictions from Bloomberg L.P. where the experiments were conducted (internship). However, we were allowed to share model predictions, as well as error analysis files at `https://github.com/UKPLab/arxiv2021-evaluation-discrepancy-nsc`.
- **Research Results**
  - All publications related to this dissertation are available in the ACL Anthology (`https://www.aclweb.org/anthology/`).
  - All research results are also documented in this dissertation itself, which is provided by the University and Regional Library Darmstadt.

According to the DFG Guidelines, the data and related software are archived internally using the infrastructure of the University and Regional Library Darmstadt, ensuring archiving for at least 10 years.

[¶] `http://dfg.de/download/pdf/foerderung/antragstellung/forschungsdaten/richtlinien_forschungsdaten.pdf`

Wissenschaftlicher Werdegang des Verfassers[¶]

09.2007 – 06.2013	Bachelor of Arts (B.A.) in Linguistic and Country Studies Belarusian State University
10.2014 – 09.2016	Master of Science (M.Sc.) in Computer Science Graduate School of Informatics, Kyoto University
10.2016 – 11.2019	Research Assistant Ubiquitous Knowledge Processing Lab, TU Darmstadt
10.2016 – today	Doctoral Researcher Ubiquitous Knowledge Processing Lab, TU Darmstadt

[¶] Gemäß §20 Abs. 3 der Promotionsordnung der TU Darmstadt

Ehrenwörtliche Erklärung[‡]

Hiermit erkläre ich, die vorgelegte Arbeit zur Erlangung des akademischen Grades "*Doctor rerum naturalium (Dr. rer. nat.)*" mit dem Titel "*Principled Approach to Natural Language Generation*" selbständig und ausschließlich unter Verwendung der angegebenen Hilfsmittel erstellt zu haben. Ich habe bisher noch keinen Promotionsversuch unternommen.

Darmstadt, den 26. April 2021

Yevgeniy Puzikov, M.Sc.

[‡] Gemäß § 9 Abs. 1 der Promotionsordnung der TU Darmstadt